

WO2003088095

Publication Title:

No title available

Abstract:

Abstract not available for WO2003088095

Data supplied from the esp@cenet database - Worldwide

Courtesy of <http://v3.espacenet.com>

(19) 世界知的所有権機関
国際事務局(43) 国際公開日
2003 年 10 月 23 日 (23.10.2003)

PCT

(10) 国際公開番号
WO 03/088095 A1(51) 国際特許分類:
H03K 19/173, H01L 21/82

G06F 17/50,

神奈川県 川崎市中原区 上小田中 4 丁目 1 番 1 号
Kanagawa (JP).

(21) 国際出願番号: PCT/JP03/04787

(22) 国際出願日: 2003 年 4 月 15 日 (15.04.2003)

(25) 国際出願の言語: 日本語

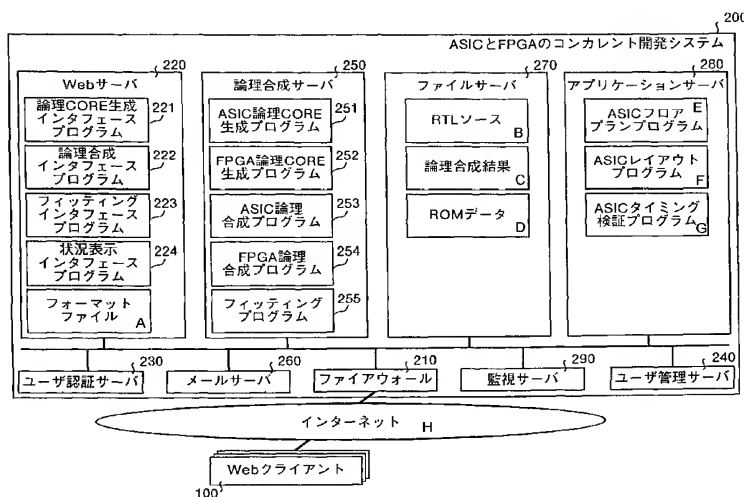
(26) 国際公開の言語: 日本語

(30) 優先権データ:
特願2002-115273 2002 年 4 月 17 日 (17.04.2002) JP
特願2002-147930 2002 年 5 月 22 日 (22.05.2002) JP(71) 出願人 (米国を除く全ての指定国について): 富士
通株式会社 (FUJITSU LIMITED) [JP/JP]; 〒211-8588

(72) 発明者; および

(75) 発明者/出願人 (米国についてのみ): 古賀 智昭
(KOGA, Chiaki) [JP/JP]; 〒812-0011 福岡県 福岡市博
多区 博多駅前三丁目 2 番 8 号 富士通九州ディ
ジタル・テクノロジー株式会社内 Fukuoka (JP). 津田 昌
行 (TSUDA, Masayuki) [JP/JP]; 〒812-0011 福岡県 福
岡市博多区 博多駅前三丁目 2 番 8 号 富士通九州
ディジタル・テクノロジー株式会社内 Fukuoka (JP). 中
山 彰二 (NAKAYAMA, Akitsugu) [JP/JP]; 〒812-0011
福岡県 福岡市博多区 博多駅前三丁目 2 番 8 号
富士通九州ディジタル・テクノロジー株式会社内
Fukuoka (JP).

[続葉有]

(54) Title: INTEGRATED CIRCUIT DEVELOPMENT METHOD, PROGRAM STORAGE MEDIUM CONTAINING
INTEGRATED CIRCUIT DEVELOPMENT METHOD, SYSTEM FOR CONCURRENT DEVELOPMENT OF ASIC AND
PROGRAMMABLE LOGIC DEVICE, DEVELOPMENT PROGRAM, AND DEVELOPMENT METHOD(54) 発明の名称: 集積回路の開発方法及び集積回路の開発方法を記憶したプログラム記憶媒体、ならびに A S I C
とプログラマブル・ロジックデバイスのコンカレント開発システム、開発プログラム及び開発方法

200. ASIC AND FPGA CONCURRENT DEVELOPMENT SYSTEM
220. Web SERVER
221. LOGICAL CORE GENERATION INTERFACE PROGRAM
222. LOGICAL SYNTHESIS INTERFACE PROGRAM
223. FITTING INTERFACE PROGRAM
224. STATE DISPLAY INTERFACE PROGRAM
A. FORMAT FILE
250. LOGICAL SYNTHESIS SERVER
251. ASIC LOGICAL CORE GENERATION PROGRAM
252. FPGA LOGICAL CORE GENERATION PROGRAM
253. ASIC LOGICAL SYNTHESIS PROGRAM
254. FPGA LOGICAL SYNTHESIS PROGRAM
255. FITTING PROGRAM
270. FILE SERVER

B. RTL SOURCE
C. LOGICAL SYNTHESIS RESULT
D. ROM DATA
280. APPLICATION SERVER
E. ASIC FLOOR PLAN PROGRAM
F. ASIC LAYOUT PROGRAM
G. ASIC TIMING VERIFICATION PROGRAM
230. USER AUTHENTICATION SERVER
260. MAIL SERVER
210. FIRE WALL
290. MONITORING SERVER
240. USER MANAGEMENT SERVER
H. INTERNET
100. Web CLIENT

(57) Abstract: An integrated circuit development method for generating a net list called a core (logical core) composed of a net connecting ports of a block not depending on a device technology by using only connection information in the block port specification which is a result of circuit architecture study and a part of logical design document, selecting blocks as objects from the core (logical core), grouping the blocks, and using the grouped core (logical core) data. A system for concurrent development of ASIC and FPGA is constituted by a fire wall for monitoring access from the Internet, a Web server for communicating with the Web client used by a user, an authentication server for performing user authentication, a user management server for managing a user, a logical synthesis server for executing an ASIC and FPGA development program, a mail server for distributing a mail to those associated with the project, a file server for storing

[続葉有]



(74) 代理人: 酒井 宏明 (SAKAI,Hiroaki); 〒100-0013 東京都千代田区霞が関三丁目2番6号 東京倶楽部ビルディング Tokyo (JP).

添付公開書類:
— 国際調査報告書

2文字コード及び他の略語については、定期発行される各PCTガゼットの巻頭に掲載されている「コードと略語のガイダンスノート」を参照。

(81) 指定国 (国内): CN, JP, KR, US.

design information, an application server for executing an ASIC implement design program, and a monitoring server for monitoring the ASCI and FPGA development state.

(57) 要約: 回路アーキテクチャ検討結果であり論理設計ドキュメントの一部であるブロックのポート仕様の接続情報のみを用いてデバイス・テクノロジーに依存しないブロックのポートとポート間を結ぶネットからなるコア（論理コア）と呼ぶネットリストを生成し、コア（論理コア）から対象ブロックを選択、グループ化し、グループ化したコア（論理コア）のデータを使用する集積回路の開発方法、およびインターネットからのアクセスを監視するファイアウォールと、ユーザが使用するWebクライアントと通信するWebサーバと、ユーザの認証を行なう認証サーバと、ユーザを管理するユーザ管理サーバと、ASICとFPGAの開発用プログラムを実行する論理合成サーバと、Projectの関係者にメールを配信するメールサーバと、設計情報を格納するファイルサーバと、ASICのインプリメント設計用プログラムを実行するアプリケーションサーバと、ASICとFPGAの開発状況を監視する監視サーバとから構成されるASICとFPGAのコンカレント開発システム。

明 細 書

集積回路の開発方法及び集積回路の開発方法を記憶したプログラム記憶媒体、な
らびにＡＳＩＣとプログラマブル・ロジックデバイスのコンカレント開発システ
ム、開発プログラム及び開発方法

技術分野

本発明は集積回路の開発方法、集積回路の開発方法を記憶したプログラム記憶
媒体及び論理合成ツールの制御装置、ならびにネットワークに接続されたコンピ
ュータからユーザが利用するＡＳＩＣとプログラマブル・ロジックデバイスのコ
ンカレント開発システム、開発プログラム及び開発方法に関する。

更に詳しくは、集積回路の開発におけるプログラマブル・ロジックデバイス構
成方法に係り、回路アーキテクチャ検討結果であり論理設計ドキュメントの一部
であるブロックのポート仕様の接続情報のみを用いてデバイス・テクノロジーに依
存しないブロックのポートとポート間を結ぶネットからなるコア（論理コア）と
呼ぶネットリストを生成し、コア（論理コア）から対象ブロックを選択、グルー
プ化し、グループ化したコア（論理コア）のデータを使用し、集積回路の開発を
行なう集積回路の開発方法、ならびにＡＳＩＣとプログラマブル・ロジックデバ
イスのシームレスなコンカレント開発を可能とし、設計品質確保と開発期間短縮
を両立するとともに、開発にかかる人的資源とコストを低減することができる
ＡＳＩＣとプログラマブル・ロジックデバイスのコンカレント開発システム、開発
プログラム及び開発方法に関する。

背景技術

集積回路の手順では、先ず仕様を決定する際、もれ等がないか慎重に検討した
後、その仕様に従い設計を行なうようになっている。第１１図は集積回路設計の
手順を示すフローチャートである。先ず、製品仕様からＡＳＩＣ（特定用途向け

I C) のスペックを取得する (g 1)。次に、回路アーキテクチャを検討する (g 2)。次に、回路アーキテクチャ検討結果に基づいて回路設計を行なう (g 3)。この回路設計は、論理検証 (g 4) を行ないながら行なう。

次に、回路設計が終了したら回路の論理合成を行なう (g 5)。論理合成が終了したら、論理合成結果に基づいてレイアウト展開する (g 6)。この段階で回路が設計できたことになる。回路が設計できたら、当該回路を製造し (g 7)、製造した回路を用いて実機の評価を行なう (g 8)。以上のシーケンスの内、本発明に係る部分は、ステップ g 5 の論理合成に係る部分である。

集積回路開発において、仕様を入力として実現する機能の検討、機能を実現する回路構成の検討を行なう回路アーキテクチャ検討を以下のような流れで実施している。通常、集積回路の開発では、製品の仕様から製品を実現する機能を洩れなく洗い出し、洗い出した機能を実現する回路の構成の検討や、IP (Intellectual Property : 知的財産) 等のマクロの検討を行なう。ここで、マクロとは IP も含む RAM、ROM 等の変更しなくても使えるものをいう。

構成を検討した回路や IP の実現規模の初期見積りを行ない、この時予め判っていればゲート数で算出し、ゲート数が判らない場合は必要な信号数と処理に必要な時間からフリップフロップ数を算出して実現規模を見積もっている。ここで、見積もった規模と各機能の入出力信号 (以降ポートと称する) 本数を基に複数の機能をグループ化し 1 つのブロックとする。このグループ化を全機能について行なう。

論理設計では、前述の機能と見積り規模を基に HDL (ハードウェア記述言語) 等の手段によりプログラマブル・ロジックデバイスを対象に回路設計を行ない、オンボードでの機能評価が行なわれる。評価完了後、ASIC 化する場合に再設計及び再検証が行なわれていた。

機能評価の完了後、ASIC 化によるコストダウンが実施される場合、プログラマブル・ロジックデバイス (例えば FPGA) を対象に設計する時点から AS

I Cを考慮した設計を施されることが少なく、I/Oバッファ、デバイス用試験回路、メモリ等のマクロなど、プログラマブル・ロジックデバイスとASIC間の違いのためASICを対象にプログラマブル・ロジックデバイスの設計データを基に再設計が発生し、再設計することによる設計データの二重管理、再設計と機能の再検証による開発期間の長期化と開発コストの増大が表面化してきている。

ここで、ASICは開発期間が長いがコストが安いという特徴があり、一方、プログラマブル・ロジックデバイス（FPGA）は開発期間は短いが高コストという特徴がある。

本発明はこのような課題に鑑みてなされたものであって、大規模ASICの開発における論理設計と、論理合成、レイアウトのコンカレント（並行した）開発に適用している回路アーキテクチャ検討において、チップを機能的に分割したブロックのポート情報と、チップのポート情報とからポート間の接続情報としてブロック間ネットリストを生成する方法（特開2000-90142号に記載）を応用した、集積回路の開発方法及び該集積回路の開発方法によって開発する集積回路を構成するブロックとブロック間ネットをブロック間ネットリストから任意の規模と個数生成するよう論理合成ツールを制御する装置を提供し、アーキテクチャの共有化を実現し、再設計と再検証を極力回避することができる集積回路の開発方法及び集積回路の開発方法を記憶したプログラム記憶媒体を提供することを目的としている。

また、近年、半導体の微細化により10Mゲートを超すASICも開発可能となっているが、電子機器の高機能化と複雑化に伴い、仕様設計、論理設計およびフロアプラン、論理合成、レイアウト設計、タイミング検証を行なうインプリメント設計が長期化すると共に設計品質の確保が困難となってきた。特に、ASIC開発のリメイクは、電子機器の開発期間の長期化のみならずコストの増大、市場投入の機会損失を招いている。

このため、開発TAT（Turn Around Time）が短く、設計変更が容易なプログ

ラマブル・ロジックデバイスが多用されつつあるが、プログラマブル・ロジックデバイスはコストがかかり、小型化が困難であることから、まずプログラマブル・ロジックデバイスで機能を実現し、プロトタイピングにてデバッグした後、量産時期にASIC化することが多い。

- 5 しかしながら、ASIC化を前提にプログラマブル・ロジックデバイスのプロトタイピングにより検証したとしても、プログラマブル・ロジックデバイスからASICへとシリアルな開発では、総開発手番を短くする事は難しいという問題がある。特に、ASIC開発時のインプリメント設計においてタイミング問題が発生すると、プログラマブル・ロジックデバイスの再設計からやり直しとなる可能性があり、半導体ベンダなど社外に依頼している場合は、依頼先の人的リソースの確保、人的リソースの確保に伴うコストの増大を招く。

- 10 更には、プログラマブル・ロジックデバイスとASICデバイス間の構造の違いからASIC専用に再設計が発生した場合は、プログラマブル・ロジックデバイスによるデバッグが無意味になるばかりでなく、開発期間が長期化してコストも増大する。どちらにしても市場投入の機会損失になる。

- 15 なお、ASICの大規模化に伴う開発期間の長期化の対策として、特開2000-90142号公報に記載のとおり、回路アーキテクチャ検討、論理設計・検証、インプリメント設計をコンカレントに行っているが、電子機器の機能の複雑化と市場の動きの早さによって、仕様設計、論理の設計、検証が長期化し開発手番を短くする事が困難となってきた。さらに、コンカレントに開発を進める場合には、ASIC開発の知識を有した人的資源と開発ツールが必要であり、半導体技術の進歩に伴い複雑化する開発ツールの教育も必要となるという問題が生ずる。

- 20 従って、この発明は、ASICとプログラマブル・ロジックデバイスのシームレスなコンカレント開発を可能とし、設計品質確保と開発期間短縮を両立するとともに、開発にかかる人的資源とコストを低減することができるASICとプログラマブル・ロジックデバイスのコンカレント開発システム、開発プログラム及

び開発方法を提供することも目的とする。

発明の開示

第 1 図は本発明方法の原理を示すフローチャートである。請求の範囲第 1 項記載の発明は、ブロックのポートとポートの接続情報なる A S I C のコア（論理コア）から、接続関係のある任意のブロックを選択しグループ化する手段により任意の規模、個数のブロックのポートとポートの接続情報からなる、論理合成ツールが読み込み可能な H D L フォーマットのコア（論理コア）を生成し（ステップ 1）、チップの端子情報から論理合成ツールで仮のチップデザインを作成し、このデザインに端子を発生させ（ステップ 2）、作成したデザイン内部にステップ 2 と同一のデザインをセルとして発生させ（ステップ 3）、デザインとセル間の同一名称のポートを接続し（ステップ 4）、接続したポート間のネットに対し、デバイス・テクノロジーに依存する I / O バッファを挿入し（ステップ 5）、ステップ 1 で作成したコア（論理コア）とセルを入れ替え、トップ階層であるデザインの階層を展開し、ネットリストを生成する（ステップ 6）、ことを特徴とする。

請求の範囲第 1 項記載の発明では、ポート名、レンジ、入出力定義からブロックのポート仕様であるエンティティを作成しファイルに出力する（ファイルに書き込む）。入出力で出力定義されているポートに関して出力信号ファイルを作成する手段により、あるブロックの入力ポートに指定された、出力元インスタンスの出力ポート名に間違いがないかチェックし、出力元インスタンス名にデバイスのパッケージの端子と接続する旨の I / O 等、キーワードの定義がある場合に、出力元インスタンスの出力ポート名にある名称をデバイスのパッケージの端子名とし、該端子が 1 本なのか複数（ベクタ）なのかをレンジの定義から判断する手段と、該端子が入力なのか出力なのかを入出力の定義から判断する手段と、該端子が双方向なのかを種別の定義から判断する手段により、コア（論理コア）のポート仕様であるエンティティを作成し、ファイルに出力する。ここで、インスタ

ンスとは回路ユニットのことである。

5 ブロックのポートが入出力定義で入力定義されている場合、該ポートの出力元
インスタンス名と出力元インスタンスの出力ポート名の組が前述の出力信号ファ
イルに存在するかチェックする手段と、前記チェック結果、インスタンス間のポ
ートの接続が可能と判断した場合、インスタンス間を接続する信号を生成し、フ
10 ァイルに出力する。全インスタンスのチェックを含め、処理が完了した時、前述
のコア（論理コア）のエンティティとインスタンス間ネットを読み込み、コア（
論理コア）を生成しファイルに出力する。

10 チェック結果が問題なければ、ブロックの入出力ポート仕様とインスタンス間
の接続ネットとインスタンスとデバイスパッケージの端子となる外部端子間の接
続ネットからなり、論理設計部分を全く持たないHDL（ハードウェア記述言語
）のファイル（以降コア（論理コア）と称する）を出力する手法をコンピュータ
15 に実行させるプログラムを記録したコンピュータが読み取り可能な記録媒体であ
る。

15 従って、請求の範囲第1項の発明のコア（論理コア）を生成する手法をコンピ
ュータに実行させるプログラムを記憶したプログラム記憶媒体によれば、集積回
路開発において、RTL設計の入力となるブロックのポート仕様の品質を事前に
確保できる効果があり、更に機能ブロックが多く存在し、設計リソースも多い大
規模集積回路の開発においても、事前にブロック間の接続が確認できていること
20 により、必ずチップが組み上がることを保証することができる。

25 請求の範囲第2項記載の発明は、ASICとプログラマブル・ロジックデバイ
スのコンカレント開発を行なう場合において、ブロックのポートとポートの接続
情報からなるASICのコア（論理コア）から、接続関係のある任意のブロック
を選択しグループ化する手段により、任意の規模、個数のブロックのポートとポ
ートの接続情報からなるコア（論理コア）を生成し（ステップ1）、チップの端
子情報から論理合成ツールで仮のチップのデザインを作成し（ステップ2）、こ
のデザインに前記チップの端子情報にある名称で端子を発生させ（ステップ3）

、この内部に前記と同一デザインをセルとして発生させ（ステップ4）、ステップ2で作成したデザインとセル間の同一名称のポートを接続し（ステップ5）、この接続ネットに対し、チップの端子情報からデバイス・テクノロジーに依存するI/Oバッファを挿入し（ステップ6）、ステップ1で生成したコア（論理コア）を前記セルと入れ替え（ステップ7）、このコア（論理コア）のトップ階層である、作成したデザインの階層を展開しプログラマブル・ロジックデバイスのチップのネットリストを生成する（ステップ8）、ことを特徴とする。

請求の範囲第2項記載の発明は、ASICとプログラマブル・ロジックデバイスのコンカレント開発において、請求の範囲第1項によるASICのコア（論理コア）から、論理合成ツールの機能を、後述する請求の範囲第3項及び請求項4記載のネットリスト生成手法をコンピュータに実行させるプログラムを記憶したプログラム記憶媒体を搭載した制御装置によって制御し、接続関係のある任意のブロックをグループ化し、プログラマブル・ロジックデバイスのコア（論理コア）として切り出し、プログラマブル・ロジックデバイスに依存するI/Oバッファを挿入し、プログラマブル・ロジックデバイスのチップのネットリストを生成することを特徴とする集積回路の開発方法である。

従って、請求の範囲第2項記載の集積回路の開発方法によれば、ASICのコア（論理コア）からプログラマブル・ロジックデバイスのコア（論理コア）としてブロック間接続を保持したまま切り出すため、本コア（論理コア）を用いたプログラマブル・ロジックデバイスの検証を実施すれば、ASICにおいては、少なくとも同一の構成での検証の重複を回避することができる。よって、ASICとプログラマブル・ロジックデバイスのコンカレント開発を効率よく進めることができる。

また、I/Oバッファを挿入する際、チップの端子情報から仮のコア（論理コア）を生成し、ブロックから生成したコア（論理コア）と入れ替える手段によって、ブロックに定義されているチップのポート情報とチップの端子情報をクロスチェックでき、ブロックのポート仕様とチップ端子仕様の両者の品質を確保する

ことができる。

請求の範囲第3項記載の発明は、ブロックのポートとポートの接続情報からなるASICのコア（論理コア）から、接続関係のある任意のブロックを選択しグループ化する手段により、任意の規模、個数のブロックのポートとポートの接続
5 情報からなるネットリストを生成することを特徴とする。

このように構成すれば、任意の規模、個数のブロックのポートとポートの接続情報からなるネットリストを生成することができる。

請求項4記載の発明は、ブロックのポートとポートの接続情報からなるASICのコア（論理コア）から、接続関係のある任意のブロックを選択しグループ化する手段により、任意の規模、個数のブロックのポートとポートの接続情報から
10 なるコア（論理コア）を生成し（ステップ1）、チップの端子情報から、論理合成ツールで仮のチップのデザインを作成し、このデザインに前記チップの端子情報にある名称で端子を発生させ（ステップ2）、この内部に前記と同一デザインをセルとして発生させ（ステップ3）、前記デザインとセル間の同一名称のポー
15 トを接続し（ステップ4）、この接続ネットに対しチップの端子情報からデバイス・テクノロジーに依存するI/Oバッファを挿入し（ステップ5）、ステップ1で生成したコア（論理コア）を前記セルと入れ替え（ステップ6）、このコア（論理コア）のトップの階層であるステップ2で作成したデザインの階層を展開してチップネットリストを生成する（ステップ7）、ことを特徴とする。

20 このように構成すれば、任意の規模、個数のブロックのポートとポートの接続情報からなるネットリストから、チップネットリストを生成することができる。

請求の範囲第5項記載の発明は、任意の規模、個数のブロックのポートとポートの接続情報からなるネットリストを生成する場合において、トップ階層のポート名をネット名から接続先のブロックのポート名と一致するようにポート名を修正し、ブロックのポートとポートの接続情報からなるネットリストを生成すること
25 とを特徴とする。

請求の範囲第5項記載の発明は、ASICとプログラマブル・ロジックデバイス

5 スのコンカレント開発において、ブロックのポートとポートの接続情報からなる
ASICのコア（論理コア）から、接続関係のある任意のブロックをグループ化
し、論理合成ツールの機能を使用してプログラマブル・ロジックデバイスのコア
（論理コア）として切り出す場合、通常切り出したコア（論理コア）のトップの
ポート名称は、当該ポートに接続されているネット名となる。

これを論理合成ツールの機能を制御してポートからコア（論理コア）内部へネ
ットをトレースし、最初に発見したブロックのポート名称に付け替えることを特
徴とする論理合成ツールの制御方法である。

10 従って、本請求の範囲第5項のプログラマブル・ロジックデバイスのコア（論
理コア）を生成する手法を論理合成ツールに実行させるプログラムを記憶した記
憶媒体によれば、ポートの機能を判断することが難しいネット名がポートの名称
となる場合の論理検証等デバッグの効率の低下を防ぐことができる。

15 上述のようにして、設計ドキュメントのデータからコア（論理コア）を生成し
、該コア（論理コア）から階層構造及び接続情報を保持した状態でプログラマブル
・ロジックデバイス用に新たにコア（論理コア）を生成することで、回路アー
キテクチャを共有することができ、回路データを埋め込み、機能検証したインス
タンス内のデバイス・テクノロジーに依存しない回路データ及びインスタンス間の
ネットは、ASIC化した場合に再検証を回避することが可能となり、ASIC
とプログラマブル・ロジックデバイス間の違いによる再設計も回避することが可
20 能となる。

また、この発明において、ブロックのポートとポート間の接続情報からなるA
S I Cのコア（論理コア）から、A S I Cを構成する任意のブロックを選択し、
グループ化することにより、選択したブロックで構成されるブロックのポートと
ポート間の接続情報からなるプログラマブル・ロジックデバイスのコア（論理コ
25 ア）の生成を制御する論理合成ツールの制御装置において、論理合成ツールが生
成した設計者が選択したブロックで構成されるプログラマブル・ロジックデバイ
スのコア（論理コア）のポートに対して、論理合成ツールがポート名として付与

したネット名を、該ポートにつながるネットを辿り、接続されているプログラマブル・ロジックデバイスを構成するブロックのポートの名称に変更する制御を行なう手段と、チップの端子情報から、チップの端子情報で指定されたポートをもつ仮のチップのデザインを論理合成ツールで生成する手段と、このデザイン内部
5 に前記仮のチップのデザインをセルとして発生させる手段と、前記仮のチップのデザインとセル間の同一名称のポート間を接続し、この接続したネットに対し、チップの端子情報からデバイス・テクノロジーに依存する I/Oバッファを挿入し、前記名称を変更した論理合成ツールが生成した、設計者が選択したブロックで構成されるプログラマブル・ロジックデバイスのコア（論理コア）とセルとを入
10 れ替える手段と、このコア（論理コア）のトップ階層を展開することによりプログラマブル・ロジックデバイスのチップネットリストを生成する手段、とを具備することを特徴とする。

このように構成すれば、プログラマブル・ロジックデバイスのチップネットリストを生成することが可能となる。

15 また、本発明は、ASICとプログラマブル・ロジックデバイスのコンカレント開発方法であって、ASICを構成する機能ブロックをポート間接続情報に基づいてグループ化し、該グループ化した機能ブロックのポートとポート間接続情報からなるネットリストをプログラマブル・ロジックデバイスのコア（論理コア）として生成するネットリスト生成工程と、前記ASICを構成する機能ブロッ
20 クの回路データからASIC用論理合成データ及びプログラマブル・ロジックデバイス用論理合成データを作成する論理合成工程と、前記グループ化した機能ブロックについてのプログラマブル・ロジックデバイス用論理合成データを前記ネットリスト生成工程により生成されたネットリストに埋め込んでプログラマブル・ロジックデバイスの回路を記録した実機評価用ROMデータを生成するROM
25 データ生成工程と、前記論理合成工程により作成されたASIC用論理合成データを用いてASICのレイアウト作成およびタイミング検証を前記ROMデータ生成工程による実機評価用ROMデータの生成とコンカレントに行なうレイアウト

ト作成工程と、前記ROMデータ生成工程により生成されたROMデータを用いた実機評価結果に基づく回路データの変更を前記ASICのレイアウト作成およびタイミング検証に反映する差分反映工程と、を含んだことを特徴とする。

また、本発明は、ASICとプログラマブル・ロジックデバイスのコンカレン
5 ト開発プログラムであって、ASICを構成する機能ブロックをポート間接続情報に基づいてグループ化し、該グループ化した機能ブロックのポートとポート間接続情報からなるネットリストをプログラマブル・ロジックデバイスのコア（論理コア）として生成するネットリスト生成手順と、前記ASICを構成する機能ブロックの回路データからASIC用論理合成データ及びプログラマブル・ロジックデバイス用論理合成データを作成する論理合成手順と、前記グループ化した機能ブロックについてのプログラマブル・ロジックデバイス用論理合成データを前記ネットリスト生成手順により生成されたネットリストに埋め込んでプログラマブル・ロジックデバイスの回路を記録した実機評価用ROMデータを生成するROMデータ生成手順と、前記論理合成手順により作成されたASIC用論理合成データを用いてASICのレイアウト作成およびタイミング検証を前記ROM
10 データ生成手順による実機評価用ROMデータの生成とコンカレントに行なうレイアウト作成手順と、前記ROMデータ生成手順により生成されたROMデータを用いた実機評価結果に基づく回路データの変更を前記ASICのレイアウト作成およびタイミング検証に反映する差分反映手順と、をコンピュータに実行させることを特徴とする。

かかる発明によれば、ASICを構成する機能ブロックをポート間接続情報に基づいてグループ化し、グループ化した機能ブロックのポートとポート間接続情報からなるネットリストをプログラマブル・ロジックデバイスのコア（論理コア）として生成し、ASICを構成する機能ブロックの回路データからASIC用
25 論理合成データ及びプログラマブル・ロジックデバイス用論理合成データを作成し、グループ化した機能ブロックについてのプログラマブル・ロジックデバイス用論理合成データを生成したネットリストに埋め込んでプログラマブル・ロジック

クデバイスの回路を記録した実機評価用ROMデータを生成し、作成したASIC用論理合成データを用いてASICのレイアウト作成およびタイミング検証を実機評価用ROMデータの生成とコンカレントに行ない、生成したROMデータを用いた実機評価結果に基づく回路データの変更をASICのレイアウト作成およびタイミング検証に反映することとしたので、ASICとプログラマブル・ロジックデバイスの効率的なコンカレント開発が可能になり、ASICの開発期間を短縮することができる。

また、本発明は、ASICとプログラマブル・ロジックデバイスのコンカレント開発システムであって、ASICを構成する機能ブロックをポート間接続情報に基づいてグループ化し、該グループ化した機能ブロックのポートとポート間接続情報からなるネットリストをプログラマブル・ロジックデバイスのコア（論理コア）として生成するネットリスト生成手段と、前記ASICを構成する機能ブロックの回路データからASIC用論理合成データ及びプログラマブル・ロジックデバイス用論理合成データを作成する論理合成手段と、前記グループ化した機能ブロックについてのプログラマブル・ロジックデバイス用論理合成データを前記ネットリスト生成手段により生成されたネットリストに詰め込んでプログラマブル・ロジックデバイスの回路を記録した実機評価用ROMデータを生成するROMデータ生成手段と、前記論理合成手段により作成されたASIC用論理合成データを用いてASICのレイアウト作成およびタイミング検証を前記ROMデータ生成手段による実機評価用ROMデータの生成とコンカレントに行なうレイアウト作成手段と、を備えたことを特徴とする。

この発明によれば、ASICを構成する機能ブロックをポート間接続情報に基づいてグループ化し、グループ化した機能ブロックのポートとポート間接続情報からなるネットリストをプログラマブル・ロジックデバイスのコア（論理コア）として生成し、ASICを構成する機能ブロックの回路データからASIC用論理合成データ及びプログラマブル・ロジックデバイス用論理合成データを作成し、グループ化した機能ブロックについてのプログラマブル・ロジックデバイス用

論理合成データを生成したネットリストに埋め込んでプログラマブル・ロジック
デバイスの回路を記録した実機評価用ROMデータを生成し、作成したASIC
用論理合成データを用いてASICのレイアウト作成およびタイミング検証を実
機評価用ROMデータの生成とコンカレントに行なうこととしたので、ASIC
5 とプログラマブル・ロジックデバイスの効率的なコンカレント開発が可能になり
、ASICの開発期間を短縮することができる。

また、本発明は、ネットワークに接続されたコンピュータからユーザが利用す
るASICとプログラマブル・ロジックデバイスのコンカレント開発システムに
おいて、前記ユーザの要求に応じてASICの論理合成を実行するASIC論理
10 合成手段と、前記ASIC論理合成手段により作成されたASICの論理合成結
果が前記ユーザの要求する速度性能を満足したか否かを判断するASIC論理合
成結果判断手段と、前記ASIC論理合成結果判断手段による判断結果に基づい
てプログラマブル・ロジックデバイスの論理合成を実行するプログラマブル・ロ
ジックデバイス論理合成手段と、前記ASIC論理合成手段によるASIC論理
15 合成の実行結果及び前記プログラマブル・ロジックデバイス論理合成手段による
プログラマブル・ロジックデバイス論理合成の実行結果を前記コンピュータに表
示する論理合成結果表示手段と、前記ASIC論理合成手段によるASIC論理
合成の実行開始と実行結果及び前記プログラマブル・ロジックデバイス論理合
成手段によるプログラマブル・ロジックデバイス論理合成の実行開始と実行結果を
20 前記ユーザに電子メールで通知する論理合成通知手段と、を備えたことを特徴と
する。

また、本発明は、ネットワークに接続されたコンピュータからユーザが利用す
るASICとプログラマブル・ロジックデバイスのコンカレント開発方法におい
て、前記ユーザの要求に応じてASICの論理合成を実行するASIC論理合成
25 工程と、前記ASIC論理合成工程により作成されたASICの論理合成結果が
前記ユーザの要求する速度性能を満足したか否かを判断するASIC論理合成結
果判断工程と、前記ASIC論理合成結果判断工程による判断結果に基づいてプ

プログラマブル・ロジックデバイスの論理合成を実行するプログラマブル・ロジックデバイス論理合成工程と、前記ASIC論理合成工程によるASIC論理合成の実行結果及び前記プログラマブル・ロジックデバイス論理合成工程によるプログラマブル・ロジックデバイス論理合成の実行結果を前記コンピュータに表示する論理合成結果表示工程と、前記ASIC論理合成工程によるASIC論理合成の実行開始と実行結果及び前記プログラマブル・ロジックデバイス論理合成工程によるプログラマブル・ロジックデバイス論理合成の実行開始と実行結果を前記ユーザに電子メールで通知する論理合成通知工程と、を含んだことを特徴とする。

かかる発明によれば、ユーザの要求に応じてASICの論理合成を実行し、作成したASICの論理合成結果がユーザの要求する速度性能を満足したか否かを判断し、判断結果に基づいてプログラマブル・ロジックデバイスの論理合成を実行し、ASIC論理合成の実行結果及びプログラマブル・ロジックデバイス論理合成の実行結果をコンピュータに表示し、ASIC論理合成の実行開始と実行結果及びプログラマブル・ロジックデバイス論理合成の実行開始と実行結果をユーザに電子メールで通知することとしたので、ユーザは、論理合成の専任者を設置する必要がなく、いつでも論理合成が実行でき、専任者が実行したが如く論理合成品質を均一に保つことができるとともに、電子メールで論理合成の開始と結果の通知を受けることができ、論理合成の進捗を定期的にコンピュータで確認する必要がなくなる。

また、本発明は、前記ユーザの要求に応じて前記ASICを構成する機能ブロックから前記ユーザが指定する複数の機能ブロックのポート間接続情報から成るネットリストを生成するネットリスト生成手段と、前記ネットリスト生成手段により生成されたネットリストに論理合成済み対象機能ブロックのデータを埋め込んでプログラマブル・ロジックデバイスの回路を記録したROMデータを生成するROMデータ生成手段と、前記ROMデータ生成手段により生成されたROMデータの生成結果を前記コンピュータに表示するROMデータ生成結果表示手段

と、前記ROMデータ生成手段により生成されたROMデータの生成結果を前記ユーザに電子メールで通知するROMデータ生成結果通知手段とをさらに備えたことを特徴とする。

この発明によれば、ユーザの要求に応じてASICを構成する機能ブロックからユーザが指定する複数の機能ブロックのポート間接続情報から成るネットリストを生成し、生成したネットリストに論理合成済み対象機能ブロックのデータを埋め込んでプログラマブル・ロジックデバイスの回路を記録したROMデータを生成し、生成したROMデータの生成結果をコンピュータに表示するとともに、ユーザに電子メールで通知することとしたので、ユーザは、プログラマブル・ロジックデバイス専用の開発環境が必要なく、プログラマブル・ロジックデバイスの回路を記録したROMデータを作成する負荷と、時間と、かかるコストとを削減することができる。

また、本発明は、前記ユーザが指定した前記ASICを構成する機能ブロックの設計が未完了で回路データが存在しない場合に、当該機能ブロックの入力端子及び出力端子に仮のフリップ・フロップなどを用いた回路を挿入したネットリストを生成する仮ネットリスト生成手段をさらに備えたことを特徴とする。

この発明によれば、ユーザが指定したASICを構成する機能ブロックの設計が未完了で回路データが存在しない場合に、当該機能ブロックの入力端子及び出力端子に仮のフリップ・フロップなどを用いた回路を挿入したネットリストを生成することとしたので、プログラマブル・ロジックデバイスのプロトタイピングによる検証において、検証対象ではない機能ブロックの設計が完了していなくとも、プロトタイピングによる検証を進めることができ、検証の効率化を図ることができる。

また、本発明は、前記ユーザが持つ最新の回路データとインプリメント設計者がインプリメント設計に取り込んでいる回路データ間の変更の規模を監視する監視手段と、前記監視手段による監視結果及びレイアウト設計にかかる時間に基づいて計画された日時に到達すると前記変更を前記ASICのインプリメント設計

への反映のタイミングにあることを前記ユーザ及び前記A S I Cのインプリメント設計者に電子メールで通知する変更タイミング通知手段と、前記変更タイミング通知手段に応答して前記ユーザが前記反映の日付を変更することで停止を要求する反映停止要求手段とをさらに備えたことを特徴とする。

- 5 この発明によれば、ユーザが持つ最新の回路データとインプリメント設計者がインプリメント設計に取り込んでいる回路データ間の変更の規模を監視し、監視結果及びレイアウト設計にかかる時間に基づいて計画された日時に到達すると変更をA S I Cのインプリメント設計へ反映するタイミングにあることをユーザ及びA S I Cのインプリメント設計者に電子メールで通知し、この通知に
10 ユーザが反映の日付を変更することで停止を要求することとしたので、発生した変更を効率よくA S I Cのレイアウト設計に反映することができるとともに、変更を反映するタイミングを設定することで、ユーザはいつまで変更することが可能かを判断することができ、早い段階でスケジュールの見直しができる。

- また、本発明は、一方では、複数の機能ブロックの全部又は一部の機能ブロッ
15 クを含むF P G Aの端子情報が記述された第1デザインと、該第1デザインの下位層として記述された該F P G Aと同様の端子情報が記述された第2デザインとから、該第1デザイン、第2デザインにおける同一端子間が接続され、かつ、該端子間にF P G A対応のバッファが挿入されたF P G A用デザイン情報を生成し、他方では、該複数の機能ブロックを含むA S I Cの端子情報が記述された第3
20 デザインと、該第3デザインの下位層として記述された該A S I Cと同様の端子情報が記述された第4デザインとから、該第3デザイン、第4デザイン間における同一端子間が接続され、かつ、該端子間にA S I C対応のバッファが挿入されたA S I C用デザイン情報を生成し、前記第2デザイン、第4デザインのそれぞれを、各デザインが含む機能ブロックの接続情報に基づいて生成された回路情報
25 で置き換える、ことでF P G A及びA S I Cのネットリストを生成することを特徴とする。

 この発明によれば、一方では、複数の機能ブロックの全部又は一部の機能ブロッ

ックを含むFPGAの端子情報が記述された第1デザインと、第1デザインの下位層として記述されたFPGAと同様の端子情報が記述された第2デザインとから、第1デザイン、第2デザインにおける同一端子間が接続され、かつ、端子間にFPGA対応のバッファが挿入されたFPGA用デザイン情報を生成し、他方

5 では、複数の機能ブロックを含むASICの端子情報が記述された第3デザインと、第3デザインの下位層として記述されたASICと同様の端子情報が記述された第4デザインとから、第3デザイン、第4デザイン間における同一端子間が接続され、かつ、端子間にASIC対応のバッファが挿入されたASIC用デザイン情報を生成し、第2デザイン、第4デザインのそれぞれを、各デザインが含む機能ブロックの接続情報に基づいて生成された回路情報で置き換える、ことで

10 FPGA及びASICのネットリストを生成することとしたので、ASICとプログラマブル・ロジックデバイスの効率的なコンカレント開発が可能になり、ASICの開発期間を短縮することができる。

15 図面の簡単な説明

第1図は、本発明方法の原理を示すフローチャートであり、第2図は、本発明の実施の形態1の動作説明図であり、第3図は、本発明の論理合成ツール制御装置の一実施の形態例を示すブロック図であり、第4図は、制御装置の一実施の形態例を示すブロック図であり、第5図は、コア（論理コア）生成プログラムを示すフローチャート（1）であり、第6図は、コア（論理コア）生成プログラムを示すフローチャート（2）であり、第7図は、本発明の実施の形態1の詳細手順を示すフローチャートであり、第8図は、コア（論理コア）デザインチェック制御を示すフローチャートであり、第9図は、グループ化制御を示すフローチャートであり、第10図は、I/Oバッファ挿入制御を示すフローチャートと処理の

20 イメージを示す図であり、第11図は、集積回路設計の手順を示すフローチャートであり、第12図は、本実施の形態2に係るASICとFPGAのコンカレント開発の概念を説明するための説明図であり、第13図は、本実施の形態2に係

25

るASICとFPGAのコンカレント開発システムのシステム構成を示す機能ブロック図であり、第14図は、ファイルサーバ内にデータを格納するためのディレクトリ構成の一例を示す図であり、第15図は、論理CORE生成用テーブルの一例を示す図であり、第16図は、本実施の形態2のASICとFPGAのコンカレント開発システムの処理手順を示すフローチャートであり、第17図は、ログイン画面の一例を示す図であり、第18図は、手順画面の一例を示す図であり、第19図は、論理CORE生成インタフェースプログラムの処理手順を示すフローチャートであり、第20図は、論理CORE生成インタフェース画面の一例を示す図であり、第21図は、ASIC論理CORE生成プログラムの処理手順を示すフローチャート（1）であり、第22図は、ASIC論理CORE生成プログラムの処理手順を示すフローチャート（2）であり、第23図は、論理COREチェックプログラムの処理手順を示すフローチャートであり、第24図は、状況表示選択画面の一例を示す図であり、第25図は、論理CORE生成状況画面の一例を示す図であり、第26図は、ASICの論理CORE生成状況表示画面での処理手順を示すフローチャートであり、第27図は、FPGA論理CORE生成インタフェース画面の一例を示す図であり、第28図は、FPGA論理CORE生成プログラムの処理手順を示すフローチャート（1）であり、第29図は、FPGA論理CORE生成プログラムの処理手順を示すフローチャート（2）であり、第30図は、互換パッケージテーブルの一例を示す図であり、第31図は、論理合成インタフェースプログラムの処理手順を示すフローチャートであり、第32図は、論理合成インタフェース画面の一例を示す図であり、第33図は、FPGA論理合成インタフェース処理手順を示すフローチャートであり、第34図は、論理合成状況表示画面の一例を示す図であり、第35図は、フィッティングデータの投入と実行画面の一例を示す図であり、第36図は、ROMデータ生成状況画面の一例を示す図であり、第37図は、予定と実績画面の一例を示す図であり、第38図は、手作業時間の一例を示す図である。

発明を実施するための最良の形態

以下、図面を参照して本発明の実施の形態例を詳細に説明する

実施の形態 1.

第 2 図は本発明の実施の形態 1 の動作説明図である。この図は、回路アーキテ
クチャ検討結果を入力とし、設計データを共通化し、回路アーキテクチャを共有
しつつ集積回路を開発するものである。

実現する機能を HDL を手段として設計する場合、製品自身であるチップは、
ある機能を持つインスタンスで構成される。このインスタンスは、HDL で設計
された複数の機能で構成されるブロックを参照するものであり、特定のブロック
がチップの機能を実現する上で必要であれば、該ブロックを複数参照するもので
ある。

第 2 図に従い本発明の流れを説明する。X、Y、Z は、回路アーキテクチャ検
討結果から作成されるブロックのポート仕様を、定義済みのフォーマットに従っ
て定義したテーブルである。これらテーブル X、Y、Z は、ブロックを設計する
際、必ず作成されるものであり、ブロック名、インスタンス名、ポート名、レン
ジ、入出力、種別、出力元インスタンス名、出力元インスタンスの出力ポート名
等からなる。これらデータは、ユーザがマニュアルで入力するものである。この
ように、各テーブルを予め作成しておくことが本発明のポイントである。

S 1 は、前述の全インスタンスのテーブルデータを入力とし、インスタンス間
及びインスタンスとデバイスパッケージの端子として定義された外部端子間が矛
盾なく接続されているかをチェックする手段と、インスタンス間のネットを定義
する手段とによりブロックのポートとブロック間接続情報のみからなるコア（論
理コア）を生成するステップである。ステップ S 1 でエラーがあった場合、エラ
ーをユーザに返し、ユーザはデータ入力をやり直す（図中に示す円状の矢印 1 は
繰り返しを示している）。

S 2 は F P G A テーブル 2 を参照し、ステップ S 1 出力のコア（論理コア）を
論理合成ツールに読み込み、プログラマブル・ロジックデバイスとするインスタ

5 ンスのグループを論理合成ツールの機能を制御して選択し、階層情報を維持した状態で新たなコア（論理コア）を生成し、デバイス・テクノロジーに依存する I/Oバッファ等を含まないステップ S 1 出力と同様のコア（論理コア）のネットリストを出力するステップである。この図で、X、Yはユーザが指定するものである。

 S 3 は、定義済みフォーマットであるデバイスパッケージの端子仕様テーブルデータの端子名から仮のコア（論理コア）を生成し、テーブルデータで指定のデバイス・テクノロジーに依存する I/Oバッファを仮のコア（論理コア）のポートに挿入後、ステップ S 2 出力のネットリストと入れ替えるステップであり、本発明のポイントである。この処理によって対象とするデバイスのチップのネットリストができた後、論理合成ツールの機能を制御して対象のデバイス・テクノロジーで論理合成済みの回路データを F P G A 合成結果のライブラリ 3 から読み込み、対応するブロックに埋め込み、所望のデバイスのネットリストを完成させるものである。F P G A として生成したネットリストは、F P G A のレイアウトツール
10 でフィッティングし、ROMデータ化される。

 また、ステップ S 2 から、A S I C 合成結果ライブラリ 4 を参照することによって A S I C 化している。

 この実施の形態 1 によれば、集積回路開発において、R T L 設計の入力となるブロックのポート仕様の品質を事前に確保できる効果があり、更に機能ブロック
20 が多く存在し、設計リソースも多い大規模集積回路の開発においても、事前にブロック間の接続が確認できていることにより、必ずチップが組み上がることを保証することができる。

 第 3 図は本発明の論理合成ツール制御装置の一実施の形態例を示すブロック図である。図において、2 2 は全体の動作を制御する制御装置、2 4 は各種情報を表示する C R T、2 1 は制御装置 2 2 に各種のコマンド等を入力する入力装置、
25 2 3 は制御装置 2 2 と接続され、各種の情報を記憶する記憶装置である。

 入力装置 2 1 は、コア（論理コア）生成プログラムの起動コマンド、論理合成

ツール制御コマンドを入力するものであり、記憶装置 2 3 には論理合成ツール、コア（論理コア）生成プログラム、論理合成制御プログラムが記憶されている。

入力装置 2 1 からブロックのテーブルファイルを指定してコア（論理コア）生成プログラム起動コマンドを入力すると、コア（論理コア）生成プログラムはテーブルファイルを読み込み、コア（論理コア）を生成し、記憶装置 2 3 にファイル出力する。この処理中にエラーがあれば、そのエラー情報を表示装置である C R T 2 4 に表示する。

設計者は、エラーがあった場合、テーブルファイルを修正後、再度コマンドを実行する。論理合成ツールの制御では、先ず制御用の必要なデータをファイルとして用意し、コア（論理コア）生成プログラムで生成したコア（論理コア）ファイルと一緒に記憶装置 2 3 内の所定の場所に格納する。そして、論理合成ツールの制御コマンドを入力装置 2 1 から入力すると、論理合成ツールが起動し、処理結果が C R T 2 4 に表示される。途中、処理に失敗した場合は、失敗した時の状態が C R T 2 4 に表示される。

第 4 図は第 3 図の制御装置 2 2 の一実施の形態例を示すブロック図である。第 3 図と同一のものは、同一の符号を付して示す。図において、3 1 は全体の動作を制御する C P U、3 2 は各種の情報を記憶するメモリ、2 1 は各種のコマンド等を入力するキーボード、2 4 は表示装置としての C R T である。2 3 は記憶装置であり、コア（論理コア）生成プログラム 3 6、論理合成ツール制御プログラム 3 5、論理合成ツール 3 4 及びオペレーティングシステム（O S）3 3 より構成されている。3 7 は各構成要素間を接続するバスである。記憶装置 2 3 としては、例えばハードディスク装置が用いられる。

このように構成されたシステムにおいて、C P U 3 1 はキーボード 2 1 からのコマンドを入力すると、記憶装置 2 3 を検索して、該当するプログラムを呼び出し、該当するプログラムを実行する。

第 5 図、第 6 図はコア（論理コア）生成プログラムを示すフローチャートである。ここでは、第 2 図の X、Y、Z のテーブルデータを使用して説明する。テー

ブルデータ X、Y、Z（インスタンスも X、Y、Z）のファイルを読み込み、1
ファイルずつ処理を行なう。F 1 で X のデータを読み込み、F 2 でポート A のテ
ーブルの全データを抽出してメモリに記憶し、このメモリのデータを参照しなが
ら次の処理を行なう。F 2' でポートは出力であるか否かをチェックする。

5 ポート A は入力であるので、インスタンス名とポート名をファイル A に出力す
るステップ F 3 はスキップし、F 3' に進む。F 3' では、パッケージ端子との
接続キーワードがあるか否かをチェックする。次に、パッケージ端子の接続キ
ーワードはないため、パッケージ端子情報をメモリに格納する F 4 はスキップし、
次のポート B の全データをメモリに記憶する。

10 ポート B は出力であるので、F 3 のステップにより “XB” を記憶装置 2 3 の
ファイル A に出力する。ここで、X はファイル名、B はポート名である。次に、
パッケージ端子の接続キーワードはないため F 4 をスキップして次のポート Z の
全データをメモリに記憶する。ポート Z は出力であるので、F 3 のステップによ
り “XZ [2 : 0]” を前記ファイル A に出力する。ここで、[2 : 0] は、2
15 、1、0 を示す。

次に、パッケージ端子の接続キーワードはないため、F 4 をスキップし、次の
ポート I の全データをメモリに記憶する。ポート I は入力であるので F 3 をスキ
ップし、次にパッケージ端子の接続キーワード “IO” があるため、“INPU
T, in, 2 : 0” をコア（論理コア）のポート名としてメモリに記憶する。次
20 に、全ポート終了であるかどうかチェックする（F 4'）。全ポートが終了して
いない場合には、ステップ F 2 に戻る。

これで、X の全ポートの処理が終了し、VHDL であれば、以下のような情報
を持ったエンティティファイルを記憶装置 2 3 に出力する。

```

A : in  std__logic ;
25  B : out std__logic ;
    Z : out std__logic vector (2 downto 0) ;
    I : in  std__logic ;

```

上記の処理を Y, Z のデータについても行ない、全ファイルが終了すると (F 5')、F 6 のステップにより VHDL であれば、以下のような情報を持ったコア (論理コア) のエンティティファイルを記憶装置 23 に出力する。

```
INPUT : in  std_logic ;
```

```
5  OUT   : out  std_logic ;
```

また、ファイル A には以下のようなデータが記録されている。

```
XB, XZ [2 : 0], YC [1 : 0], YO, ZF
```

次に、F 7 のステップで上記ファイル A のデータを読み込み、F 8 のステップでメモリに記憶し、1 ファイルずつ処理を行なう。F 8' のステップでは、ポートは入力であるかどうかチェックする。ここで、X のテーブルデータを例に説明する。ポート A は入力であるので、F 9 のステップにより出力元の Y とポート名の C を抽出し、続く F 10 のステップにより YC [1 : 0] を F 8 で記憶したデータで検索する。そして、F 10' で一致するかどうかチェックする。

検索の結果、存在するので、F 12 のステップにより以下のような接続情報をファイル B に出力する。結果が一致しない場合は、ポートの名称又はレンジが異なっている等のエラー情報を記憶装置 23 のログファイルに出力すると共に、CRT 24 にも表示する。

```
A=>YC
```

次に、同様にしてポート B は出力であるので、F 11 のステップにより、以下のような接続情報をファイル B に出力する。

```
B=>XB
```

次に、同様にしてポート Z は出力であるので、F 11 のステップにより、以下のような接続情報をファイル B に出力する。

```
Z=>XZ
```

次に、ポート I は入力であるが、F 10 のステップの検索で "IOINPUT" が発見できず、F 13 のステップでエラーとしてログファイルに出力されると共に、CRT 24 にも表示する。但し、パッケージの端子と接続されるポートの

場合は問題ではなく、逆にログに出力されたエラー情報からどのインスタンスのどのポートがパッケージのどの端子と接続したかの確認ができる。次に、F 1 2のステップにより、以下のような接続情報をファイルBに出力する。

I => I O I N P U T

- 5 ステップF 1 2' では、全テーブルの全ポートが終了したかどうかをチェックし、終了していない場合には、ステップF 8' に戻る。

10 以上の処理をY, Zのテーブルデータについても行ない、終了すると、F 1 4のステップにより、接続情報が作成できなかったポートをログファイルに出力し、F 1 5のステップによりF 6のステップで出力したコア（論理コア）のエンティティリストとF 1 1とF 1 2のステップで出力した接続情報を結合し、コア（論理コア）のネットリストを生成し、記憶装置23に出力してプログラムを終了する。

15 次に、図を用いて本発明の実施の形態1の詳細手順について説明する。第7図は本発明の実施の形態1の詳細手順を示すフローチャートである。このフローチャートは、第2図で説明した実施の形態1のうち、S 1で出力したコア（論理コア）を入力とするS 2以降の処理を詳細に示すものである。図中、実線は処理の流れを示し、破線はデータの流れを示す。処理の実行前に、入力装置を操作してステップS 2で出力したコア（論理コア）と必要があれば、以降の説明で出てくる制御ファイルを制御プログラムが参照する記憶装置内の場所へ複写等の手段によって準備する。

20 先ず、コア（論理コア）デザインをチェックする（a 1）。そして、該コア（論理コア）デザインがOKかどうかチェックする（a 1'）。OKでない場合には、第2図説明のテーブルデータをチェック後、再実行する（a 1 0）。OKである場合、ファイル12、13を参照してグループ化し（a 2）、コア（論理コア）10として格納する。次に、端子名とバッファ名からなるテーブル14を参照してダミーコア（論理コア）を生成する（a 3）。この結果、ダミーコア（論理コア）11が生成される。

次に、このダミーコア（論理コア）11及びテーブル14を基にしてI/Oバッファを挿入する（a4）。次に、コア（論理コア）10とダミーコア（論理コア）11とを比較チェックする（a5）。そして、OKであるかどうかチェックする（a5'）比較の結果、OKである場合には終端処理を行なう（a6）。OKでない場合には、デバイスパッケージの端子仕様をチェックし（a9）、第2図説明のテーブルデータをチェック後、再実行する（a10）。

ステップa6の終端処理が終了したら、合成結果ファイル15を参照して回路データを埋め込み（a7）、ファイル16を参照してDFT回路を挿入する（a8）。

次に、第7図で説明した各ステップの更に詳細な説明をする。第8図はコア（論理コア）デザインチェック制御を示すフローチャートである。先ず、コア（論理コア）のファイルがあるかどうかチェックし（a11）、ある場合には、論理合成ツールで読み込ませる（a12）。ない場合には、何もしない。ステップa1（第7図参照）では、ステップS2で出力したコア（論理コア）を入力とし、コア（論理コア）のHDL記述文法、空き入力ポート等をチェックするステップで、コア（論理コア）のファイルが存在すれば、ステップa12で論理合成ツールにコア（論理コア）を読み込ませる。論理合成ツールは、読み込んだ際、文法等のチェックを行なう機能を持っており、結果がエラーであればCRTで内容を確認し、ステップa10に示す通りブロックのポート仕様を記述したテーブルデータを見直し、再度デザインチェックを実行する。

第9図はグループ化制御を示すフローチャートで、第7図のステップa2の詳細を示している。先ず、コア（論理コア）のファイルがあるかどうかチェックし（a13）、ある場合にはグループ化するインスタンスのファイルがあるかどうかチェックする（a14）。ある場合には、即ち、ステップa1の結果に問題がない場合、プログラマブル・ロジックデバイス化する対象のブロック名を1行に1ブロックずつ記載されたファイル13があればそれを読み込み（a21）、対象のブロックの階層情報を維持した状態で新たなコア（論理コア）として論理合

成ツールのグルーピング機能を制御してグループ化する（a 2 2）。

- 5 グルーピング機能を使用する場合、コア（論理コア）のポート名がインスタンス間を接続しているネット名となるため、コア（論理コア）の全てのポートに対してコア（論理コア）内部へネットをトレースし、最初に発見したインスタンスのポート名でコア（論理コア）に新規ポートを発生させネットの名前となっているコア（論理コア）のポートに接続されているネットを発生させた新規ポートに接続し、ネット名となっているポートを削除する（a 2 3）。新しいポートを作成して古いポートを消すためである。

- 次に、モニタポート情報があるかないかチェックする（a 2 3'）。ある場合には、コア（論理コア）に指定のインスタンスのポート名でポートを発生させ、
10 ブロックの端子と接続する（a 2 4）。ない場合には、デバイス・テクノロジーに依存する I/Oバッファ等を含まないネットリストを記憶装置 10（第 7 図参照）に出力する（a 2 5）。この場合、表示装置においては、論理合成ツールの表示機能を使用して確認することができる。

- 15 このステップ a 2 では、他に必要があれば、グループ化したブロックのポートをコア（論理コア）のポートとして発生させる（a 2 4）こともできる。この機能は、プログラマブル・ロジックデバイスの機能評価時にモニタしたいポートがある場合に使用する。指定方法は、ファイル 1 2 に示すように、ポート名を 1 行に 1 ポートずつ記載したファイルをグルーピングとポート名称変更後に読み込ませることで行なう。
20

- ここまでは、設計者自身がブロックのポート仕様を決めるデータを用いてコア（論理コア）を生成するため、コア（論理コア）内のブロック間のポート接続に間違いはないはずである。しかしながら、ステップ S 1（第 2 図参照）で生成するコア（論理コア）をそのまま使用する A S I C の場合、コア（論理コア）内ブロックのテーブルデータに定義するデバイスパッケージの端子名称はデバイスパッケージの端子仕様を参照するのが普通であり、従って単純なミスやプリント基板設計からの変更等で相違がある可能性がある。これとは逆に、ステップ a 2（
25

第7図参照)で生成したコア(論理コア)のポートがデバイスパッケージの端子となる場合、該ポートが仕様となるため、前述と同様に相違がある可能性がある。この問題をステップa 3からa 5のステップで解消している。

5 第10図はI/Oバッファ挿入制御を示すフローチャートと処理のイメージを示す図である。このフローチャートは、第7図のステップa 4の処理を示している。以下の説明では、a 3からa 5は別としているが、処理は1つのコマンドで一連の処理である。

10 先ず、外部端子情報及びコア(論理コア)のファイルがあるかどうかチェックする(a 1 5)。ある場合には、以下の処理を行なう。a 3では、a 2と違い第7図の1 4で示すように第1カラムに端子名称、第2カラムに使用するデバイステクノロジーに依存するI/Oバッファ名を定義したデバイスパッケージの端子仕様のテーブルデータが存在すれば、それを読み込ませ、ダミーのチップのデザイン(第1カラムの名称の端子情報を記述した第1デザイン)を作成し(a 3 1)、この内部に前記ダミーチップのデザインをセル(第1デザインの下位層として

15 記述した第1デザインと同様の端子情報を記述した第2デザイン)として発生させ、前記デザインと前記セル間の同一名称のポートを接続し、記憶装置11にダミーのコア(論理コア)を出力する。

20 ステップa 4では、記憶装置11のダミーのコア(論理コア)を読み込み、デバイスパッケージの端子仕様テーブルデータ14の第2カラムにあるI/Oバッファをダミーのコア(論理コア)のネットに対しチップの端子情報からデバイス・テクノロジーに依存するI/Oバッファを挿入する(a 3 2)。

25 ステップa 5では、上記ダミーコア(論理コア)のポートのうち機能に関係のない論理に係わらないデバイス特有でテスト用等のポートを削除し、内部のセルとして記憶装置10のコア(論理コア)と入れ替え(a 3 3)、端子名の不一致があるかどうかチェックする(a 3 3')。2つのコア(論理コア)間のポート名が全て一致すれば、埋め込みに成功である。こうすることで、デバイスパッケージの端子仕様テーブルデータの端子名称とコア(論理コア)のポート名称をク

ロスチェックでき、上記の問題を解消できる。ある場合は処理は終了し、ない場合にはネットリストを出力する。

失敗した場合には、ステップ a 9 に示す通り、デバイスパッケージの端子仕様をチェックし、a 3 のステップから再実行するか、ステップ a 10 に示す通りブロックの端子仕様を定義したテーブルデータをチェックし、ステップ S 1（第 2 図参照）から再実行する。成功した場合、コア（論理コア）の階層を展開することでチップと化す。

ステップ a 6 では、前記チップの全ブロックの入出力ポートにフリップフロップをブロック内部に接続し、終端する。この終端処理は、仮に回路設計が完了していないブロックが存在したとしても問題なくネットリストを生成し、レイアウト作業を可能とするためのものである。

ステップ a 7 では、論理合成済みの回路データを合成結果のライブラリから読み込み、対応するブロックに埋め込む。この時、チップ内部のブロックに対する合成結果が存在せず、且つ該ブロックのデザイン名称又はセル名称が名前付け規則に従ったものであれば、該ブロックの入力ポートと出力ポートを予め規則に従って該ブロック内部で接続処理する。

この処理は、プログラマブル・ロジックデバイスの場合には、存在せず、プログラマブル・ロジックデバイスの D L L 等特有のブロックを A S I C とする場合に施す。このブロックを削除せずブロック内部で接続する処理によって、アーキテクチャが保持できる。また、ブロック内部にメモリ等のマクロが存在する場合は、デバイス特有のメモリの回路データに入れ替える。

a 8 は A S I C の場合の処理で、デバイスをテストする S C A N テスト回路の自動挿入、又は S C A N テスト回路を接続するブロックの順序を定義したファイル 1 6 を入力し、定義順に接続する。

以上、述べたように、本実施の形態 1 によれば、設計ドキュメントからのデータからコア（論理コア）を生成し、このコア（論理コア）から階層構造及び接続情報を保持した状態でプログラマブル・ロジックデバイス用に新たにコア（論理

コア) を生成することによって、回路アーキテクチャを共有できる効果があり、このコア (論理コア) に対して回路データを埋め込み機能を検証したインスタンス内のデバイス・テクノロジーに依存しない回路データ及びインスタンス間のネットは、ASIC化した場合に再検証を回避することが可能となる。よって、ASICとプログラマブル・ロジックデバイスのコンカレント開発を効率よく進めることを可能にできる効果もある。

実施の形態 2.

次に、本発明に係るASICとFPGAのコンカレント開発システムについて詳細に説明する。なお、論理の設計言語には、C言語、UMLなどがあるが、本実施の形態 2 では、HDLを設計言語として説明する。

まず、本実施の形態 2 に係るASICとFPGAのコンカレント開発の概念について説明する。第 12 図は、本実施の形態 2 に係るASICとFPGAのコンカレント開発の概念を説明するための説明図である。同図に示すように、このASICとFPGAのコンカレント開発の特徴は、FPGAによるプロトタイプ検証をASICのインプリメント設計とコンカレントに進めるために、プロトタイプ検証に必要なFPGAの回路を記録したROMデータをASICのインプリメント設計から提供し、FPGAとASIC開発をシームレスにできることである。

このASICとFPGAのコンカレント開発では、回路アーキテクチャ検討においてFPGAによるプロトタイプ検証を考慮し、適当な予想規模の実現回路での実現機能の分割、ASICとFPGA間の構造的違いに係る機能の階層化などを行ない、ASICとFPGA設計間の共通の機能ブロック構成及び機能ブロックのポート仕様を作成する。この機能ブロック構成及び機能ブロックのポート仕様データは、ASICのインプリメント設計のフロアプラン、論理合成、レイアウト設計の共通のデータとなる。

RTL設計・検証では、回路アーキテクチャ検討結果の構造に従ってASICとFPGA共通のRTL設計を行ない、機能毎にコーナーケースを重点に論理検

証ツールを用いてソフト検証を行なう。このRTL設計・検証とコンカレントにASICの論理合成を実施し、ASICとしての特性が確保できれば、次にFPGAの論理合成を実施し、逐次プロトタイピング用基板上のFPGAに回路を形成し、FPGAのプロトタイピングによる検証を可能にする。

- 5 一方、ASICのレイアウトでは、ASICとFPGA設計間の共通の機能ブロック構成を基に、ソフト検証とプロトタイピングによる検証結果を随時コンカレントに反映させることによって、FPGAのプロトタイピングによる検証の完了からASIC開発完了までの期間を短縮する。

- 次に、本実施の形態2に係るASICとFPGAのコンカレント開発システムのシステム構成について説明する。第13図は、本実施の形態2に係るASICとFPGAのコンカレント開発システムのシステム構成を示す機能ブロック図である。同図に示すように、このASICとFPGAのコンカレント開発システム200は、ファイアウォール210と、Webサーバ220と、ユーザ認証サーバ230と、ユーザ管理サーバ240と、論理合成サーバ250と、メールサーバ260と、ファイルサーバ270と、アプリケーションサーバ280と、監視サーバ290とを有する。また、このASICとFPGAのコンカレント開発システム200は、インターネットを介してWebクライアント100から利用することができる。
- 10
- 15

- ファイアウォール210は、インターネットからのアクセス要求に対し、設定された通信手順に従ったアクセス要求のみを受入れるコンピュータであり、ASICとFPGAのコンカレント開発システム200への外部からの不正なアクセスを防ぐ。
- 20

- Webサーバ220は、インターネットを通じて成されるWebクライアント100からの要求に対して情報送信を行なうコンピュータであり、ASIC又はFPGAを構成する機能ブロックのポートと機能ブロックのポート間の接続情報のみから論理COREを生成する論理CORE生成インタフェースプログラム221、論理合成インタフェースプログラム222、フィッティングインタフェー
- 25

スプログラム 223、状況表示インタフェースプログラム 224、論理 CORE 生成に必要なフォーマットファイルと論理合成に必要なフォーマットファイルとを有し、このコンピュータが Web クライアント 100 からの要求に応じてこれらのプログラムを実行し、Web クライアント 100 に結果を送信する。

5 ユーザ認証サーバ 230 は、ユーザの認証を行なうコンピュータであり、契約に基づいて使用するユーザ名とパスワードが登録されている。ユーザ管理サーバ 240 は、ユーザの登録、削除を行なうコンピュータであり、契約に基づいて使用するユーザ名と後述する Project 名とメールアドレスが登録されている。

10 論理合成サーバ 250 は、ASIC 論理 CORE 生成プログラム 251 と、FPGA 論理 CORE 生成プログラム 252 と、ASIC 論理合成プログラム 253 と、FPGA 論理合成プログラム 254 と、FPGA のレイアウトであるフィッティングプログラム 255 とを有するコンピュータであり、この論理合成サーバ 250 にある論理合成及びフィッティングのプログラムは、論理合成インタフェースプログラム 222 から起動され、ファイルサーバ 270 に格納されている
15 RTL ソースを読み込み、ASIC 及び FPGA の論理合成と FPGA のフィッティングを実行する。

メールサーバ 260 は、メール送受信ソフトウェアを有するコンピュータであり、Web サーバ 220 から実行された処理情報、Web クライアント 100 からの情報をユーザとシステム内へメール配信する。
20

ファイルサーバ 270 は、論理合成対象 RTL ソース、論理合成結果、FPGA の ROM データなどを格納する記憶媒体である。第 14 図は、ファイルサーバ内にデータを格納するためのディレクトリ構成の一例を示す図である。同図に示す Project 41 とは、ASIC 及び FPGA を開発する Project 名
25 あるいは ASIC のニックネームのディレクトリであり、この名称は契約に基づいて設定される。

IO42 は、論理 CORE を生成するためのデータを格納するディレクトリで

あり、ASICを構成する機能ブロック単位に、第15図に示す機能ブロックのポート仕様を記載した論理CORE生成用テーブルのファイルを格納するディレクトリASIC48と、論理CORE生成インタフェースプログラム221がユーザの指定に従ってディレクトリASIC48内の論理CORE生成用テーブルのファイルを複写して格納するディレクトリFPGA49とをもつ。また、ディレクトリFPGA49の下に、ユーザが指定するFPGAの番号の数のディレクトリが存在する。なお、論理CORE生成用テーブル及び論理CORE生成インタフェースプログラム221の詳細については後述する。

CORE43は、論理CORE生成プログラムが生成した論理COREを格納するディレクトリであり、ASICを構成する機能ブロックのポート仕様から生成したASICの論理COREを格納するディレクトリASIC50と、FPGAを構成する機能ブロックのポート仕様からFPGAの論理COREを格納するディレクトリFPGA51をもつ。

RTL44は、後述する論理合成インタフェースからユーザがアップロードするHDL（ハードウェア記述言語）で表現された回路設計データ（以降RTL）を格納するディレクトリであり、ASICの論理COREを構成する機能ブロック単位に格納するディレクトリASIC52と、ユーザの指定に従ってASICディレクトリ内のRTLを複写して格納するディレクトリFPGA53とをもつ。

SYNTHESIS45は、論理合成サーバ250がRTLを論理合成した結果を格納するディレクトリであり、ASICの論理合成結果をASICの論理COREを構成する機能ブロック単位に格納するディレクトリをもつディレクトリASIC54と、それぞれのFPGAの論理合成結果を格納するディレクトリをもつディレクトリFPGA55をもつ。

ROM46は、FPGAの論理合成結果を基にFPGAのレイアウトであるフィッティング後生成されるFPGAの回路データを記憶するROMデータを格納するディレクトリであり、FPGA毎に格納するディレクトリをもつ。LAYO

UT 47は、ASICのレイアウト設計担当者がASICのレイアウト設計を行なう作業ディレクトリである。

アプリケーションサーバ280は、ASICのフロアプラン、レイアウト設計、タイミング検証を行なうプログラムを有するコンピュータである。このコンピュータでインプリメント設計者がASICのフロアプラン、レイアウト設計、タイミング検証を行なう。

監視サーバ290は、ユーザ管理サーバ240に登録されているユーザ名とProject名を取得し、それぞれのProject名のディレクトリ内にあるデータにおいて、インプリメント設計者が管理するディレクトリLAYOUT 47内のデータとその他のディレクトリ内のデータとの比較、論理合成、レイアウト処理にツールが要した時間の収集を1日に1回ずつ2回に分けて行ない、1回目がインプリメント設計者への変更内容の通知を行ない、2回目に収集した時間を基に変更を反映するスケジュールを更新するコンピュータである。

次に、本実施の形態2のASICとFPGAのコンカレント開発システム200の処理手順について説明する。第16図は、本実施の形態2のASICとFPGAのコンカレント開発システム200の処理手順を示すフローチャートである。

同図に示すように、このASICとFPGAのコンカレント開発システム200にWebクライアント100を用いてアクセスすると、Webサーバ220は、第17図に示すログイン画面の表示制御データをWebクライアント100へ送り、Webクライアント100は、受取ったデータに基づいて画面を表示する。ユーザは、このログイン画面で契約に基づいて登録しているユーザ名とパスワードを入力してLoginボタンを押すと、Webクライアント100は、ユーザ名とパスワードをWebサーバ220に送る。

Webサーバ220は、受取ったユーザ名とパスワードをユーザ認証サーバ230に問い合わせる。ユーザ認証サーバ230は、ユーザ名とパスワードが登録されているか否かを確認し（ステップS501）、その結果をWebサーバ22

0に返す。Webサーバ220は、ユーザ認証サーバ230から拒絶された場合、ログイン拒否画面の表示制御データをWebクライアント100へ送り、Webクライアント100は、受取ったデータに基づいてログイン拒否を画面に表示して終了する（ステップS502）。

- 5 Webサーバ220は、ユーザ認証サーバ230からログインを受け付けられた場合、第18図に示す手順画面の表示制御データをWebクライアント100へ送り、Webクライアント100は受取ったデータに基づいて手順画面を表示する（ステップS503）。

- ユーザは、第18図に示した手順画面に従いメニューを選択し、ASICとFPGAのコンカレント開発を行なう。ユーザが「フォーマット1」及び「フォーマット2」を選択すると（ステップS504の肯定）、Webサーバ220内にある対象の設計に必要なフォーマットファイルがWebクライアント100にダウンロードされる（ステップS505）。

- 「フォーマット1」は、第15図に示した論理CORE生成用テーブルのフォーマットデータである。「フォーマット2」は、後述するASICの論理合成で使用するASICのパッケージの端子名称と端子番号割り付け、及びASICと外部デバイス間を電氣的にインタフェースするIOバッファを定義したテーブルのフォーマットデータである。

- ユーザが論理CORE生成を選択すると（ステップS506の肯定）、Webクライアント100は論理CORE生成が選択された事をWebサーバ220へ送り、Webサーバ220は論理CORE生成インタフェースプログラム221を起動する（ステップS507）。

- ユーザが論理合成を選択すると（ステップS508の肯定）、Webクライアント100は論理合成が選択された事をWebサーバ220へ送り、Webサーバ220は論理合成インタフェースプログラム222を起動する（ステップS509）。

ユーザがフィッティングを選択すると（ステップS510の肯定）、Webク

クライアント100はフィッティングが選択された事をWebサーバ220へ送り、Webサーバ220はフィッティングインタフェースプログラム223を起動する（ステップS511）。

5 ユーザが状況表示を選択すると（ステップS512の肯定）、Webクライアント100は状況表示が選択された事をWebサーバ220へ送り、Webサーバ220は状況表示インタフェースプログラム224を起動する（ステップS513）。

次に、論理CORE生成インタフェースプログラム221の処理手順について説明する。第19図は、論理CORE生成インタフェースプログラム221の処理手順を示すフローチャートである。同図に示すように、この論理CORE生成
10 インタフェースプログラム221は、第20図に示す論理CORE生成インタフェース画面の表示制御データをWebサーバ220へ送り、Webサーバ220は受取った論理CORE生成インタフェース画面の表示制御データをWebクライアント100へ送り、Webクライアント100は受取った論理CORE生成
15 インタフェース画面の表示制御データに基づいて論理CORE生成インタフェース画面を表示する（ステップS801）。

ユーザは、Project名を入力して実行ボタンを選択し、Webクライアント100によって表示されるファイル選択画面に従ってASICの論理COREを生成するためのASICを構成する機能ブロックのIO仕様を定義した第1
20 5図に示す論理CORE生成用テーブルのファイルを指定する。

Webクライアント100は、Projectの指定データとIOの論理CORE生成用テーブルのファイルをWebサーバ220へ送り、Webサーバ220は、ファイルサーバ270内に、第14図に示したIO42, CORE43, RTL44, SYNTHESIS45, ROM46, LAYOUT47などのディレクトリを作成し（ステップS802）、IO42下のディレクトリASIC
25 48に論理CORE生成用テーブルのファイルを格納し、Project名を指定し、ASIC論理CORE生成プログラム251を実行し、実行プロセスID

をメモリに格納する（ステップS 8 0 3）。

ここで、A S I C論理CORE生成プログラム2 5 1の処理手順について説明する。第2 1図及び第2 2図は、A S I C論理CORE生成プログラム2 5 1の処理手順を示すフローチャートである。第2 1図に示すように、このA S I C論理CORE生成プログラム2 5 1は、第1 5図に示した論理CORE生成用テーブルX、Y、Zを、第1 4図のディレクトリA S I C 4 8から読み込み、1ファイルずつ処理を行なう。

論理CORE生成用テーブルX、Y、Zは、回路アーキテクチャ検討結果から作成される機能ブロックのポート仕様を、定義済みの「フォーマット1」に従って定義したテーブルである。このテーブルは、機能ブロックを設計する際必ず作成されるものであり、機能ブロック名、インスタンス名、ポート名、レンジ、入出力、種別、出力元インスタンス名、出力元インスタンスの出力ポート名などから成る。インスタンスとは、A S I Cを構成する機能ブロックを参照するものであり、同一機能の機能ブロックを複数使用する場合はこのインスタンスの名称を変えて組込まれる。

まず、Xのデータを読み込み（ステップS 1 0 0 1）、ポートAのテーブルの全データを抽出してメモリに格納し（ステップS 1 0 0 2）、ポートAが出力か否かを調べる（ステップS 1 0 0 3）。その結果、ポートAは入力であるので、次にパッケージ端子の接続キーワード” I O”の有無を調べる（ステップS 1 0 0 5）。その結果、接続キーワード” I O”はないので、次のポートBの全データをメモリに格納する。

ポートBは出力であるので、” X B”を第1 4図のディレクトリA S I C 5 0にAの名称でファイル出力する（ステップS 1 0 0 4）。次にパッケージ端子の接続キーワード” I O”はないので、次のポートZの全データをメモリに格納する。ポートZは出力であるので、” X Z [2:0] ”をファイルAに出力する。次にパッケージ端子の接続キーワード” I O”はないので、次のポートIの全データをメモリに格納する。

ポート I は入力であり、次にパッケージ端子の接続キーワード” I O” があるので” I N P U T, in, 2 : 0” を論理 C O R E のポート名としてメモリに格納する（ステップ S 1 0 0 6）。これで X の全ポートの処理が終了し（ステップ S 1 0 0 7 の肯定）、H D L の文法に従ってポート名と入出力とレンジ情報を定義したエンティティファイルを第 1 4 図のディレクトリ A S I C 5 0 に X の名称でファイル出力する（ステップ S 1 0 0 8）。出力される内容は、V H D L であれば以下のようなものである。

```

A : in std_logic ;
B : out std_logic ;
Z : out std_logic_vector (2 downto 0) ;
I : in std_logic ;

```

そして、上記の処理を Y、Z のデータについても行なう。また、この段階でファイル A には次のようなデータが記録されている。

```

XB, XZ[2 : 0], YC[1 : 0], YO, ZF

```

上記 X、Y、Z の全ファイルの処理が終了すると（ステップ S 1 0 0 9 の肯定）、上記ファイル A（ステップ S 1 0 1 0）のデータを読み込み、メモリに格納し（ステップ S 1 0 1 1）、1 ファイルずつ処理を行なう。ここでは、X のテーブルデータを例に説明する。まず、ポート A が入力であるか否かを調べ（ステップ S 1 0 1 2）、その結果、ポート A は入力であるので、出力元の Y とポート名の C を抽出する（ステップ S 1 0 1 3）。そして、メモリを Y C [1 : 0] で検索し（ステップ S 1 0 1 4）、一致するデータの有無を調べる（ステップ S 1 0 1 6）。その結果、一致するデータがメモリに存在するので、以下のような接続情報を第 1 4 図のディレクトリ A S I C 5 0 に B の名称でファイル出力する（ステップ S 1 0 1 8）。結果が一致しない場合は、ポートの名称またはレンジが異なっているなどのエラー情報をディレクトリ A S I C 5 0 のログファイルに出力する（ステップ S 1 0 1 7）。

```

A => YC

```

同様にして、ポートBは出力であるので、以下のような接続情報をディレクトリAS I C 5 0のファイルBに出力する（ステップS 1 0 1 5）。

B => X B

同様にして、ポートZは出力であるので、以下のような接続情報をディレクトリAS I C 5 0のファイルBに出力する。

Z => X Z

同様にして、ポートIは入力であり、検索で” I O I N P U T” が発見できないので、エラーとしてディレクトリAS I C 5 0のログファイルに出力する。ただし、パッケージの端子と接続されるポートの場合は問題ではなく、逆にログに出力されたエラー情報からどのインスタンスのどのポートがパッケージのどの端子と接続したかの確認ができる。次に、以下のような接続情報をディレクトリAS I C 5 0のファイルBに出力する。

I => I O I N P U T

以上の処理をY,Zのテーブルデータについても行って終了すると（ステップS 1 0 1 9の肯定）、接続情報が作成できなかったポートをログファイルに出力し（ステップS 1 0 2 0）、VHDLであれば、以下のような情報をもつたAS I Cの論理COREのエンティティファイルをディレクトリAS I C 5 0に出力する（ステップS 1 0 2 1）。

I N P U T : in std_logic ;

20 O U T : out std_logic ;

また、論理COREのエンティティファイルと接続情報を結合し、AS I Cの論理COREをディレクトリAS I C 5 0に第17図で指定されたP r o j e c t名でファイル出力して処理を終了し、第19図に示した論理CORE生成インタフェースプログラム221に戻る（ステップS 1 0 2 2）。

25 論理CORE生成インタフェースプログラム221は、ディレクトリAS I C 5 0内のログファイルにエラーがないかを検索する。もしエラーが記録されていれば（ステップS 8 0 4の肯定）、エラーを読み込み（ステップS 8 0 5）、P

project名と、ユーザ管理サーバ240にProject名で問い合わせ得たユーザのメールアドレスと、エラー情報と、プロセスIDとをメールサーバ260へ送り、メールサーバ260からユーザへメールを送信する（ステップS806）。ユーザは、このメールでエラー内容を確認し、エラーがなくなるまで

5 論理CORE生成の処理を繰り返す。

一方、ログファイルにエラーが記録されていなければ（ステップS804の否定）、Project名と、ユーザ管理サーバ240にProject名で問い合わせ得たユーザのメールアドレスと、プロセスIDと、論理CORE生成終了メッセージとをメールサーバ260へ送り（ステップS807）、メールサーバ260からユーザへメールを送信する。

10

また、ディレクトリASIC50に”CORE+プロセスID+日時”の名称でディレクトリを作成し、ディレクトリASIC50にProject名で生成した論理COREとログファイルとをこのディレクトリに移動し、このディレクトリにIOの名称でディレクトリを作成し、論理COREの生成に使用したディレクトリASIC48にあるテーブルデータをこのIOディレクトリに移動し、

15 論理合成サーバ250内の論理COREチェックプログラムにこのProject名と”CORE+プロセスID+日時”の名称で作成した論理COREのディレクトリ名を指定し、論理COREチェックプログラムを起動する（ステップS808）。

20 この論理COREチェックプログラムは、ASICの論理COREの文法チェック、論理COREを構成する各インスタンスの入出力ポートの未接続チェックを行ない、レポートを出力するように論理合成ツールのコマンドが記述されている。

ここで、論理COREチェックプログラムの処理手順について説明する。第2

25 3図は、論理COREチェックプログラムの処理手順を示すフローチャートである。この論理COREチェックプログラムは、受取ったProject名のディレクトリ下であり、第14図のディレクトリASIC50に対象とするASIC

の論理COREのディレクトリがあるかを調べ（ステップS1201）、もしあれば、その論理COREのディレクトリがあるASICディレクトリ内にWORKディレクトリを作成し（ステップS1202）、論理合成ツールにチェックを実行させる（ステップS1203）。

- 5 実行が終了すると、WORKディレクトリ内にあるレポートファイル内からエラー情報を抽出し、ユーザ管理サーバ240にProject名で問い合わせ得たユーザのメールアドレスと論理COREのディレクトリ名から抽出したプロセスIDと共にメールサーバ260へ送り、メールサーバ260からユーザへメール送信する（ステップS1204）。ユーザはこの送られたメールの内容でエラー情報が意図したものか確認する。

- 10 この一連のフローで生成された論理COREは、ユーザの論理設計においてASICを構成するインスタンス間の結線ミスが無い限り必ずASICとして組み上がることを保証するものである。この効果は、論理検証にて現れる。ASICを構成する複数のインスタンスの機能検証において期待と違った動作をした場合、
- 15 インスタンス間の接続は保証できていることからインスタンスを構成する各機能に絞ったデバッグが可能となる。

- 次に、ユーザが第18図の手順画面において状況表示を選択した場合の処理について説明する。ユーザが第18図の手順画面において状況表示を選択すると、状況表示インタフェースプログラム224が起動される。この状況表示インタフェースプログラム224は、第24図に示す状況表示選択画面の表示制御データをWebサーバ220へ送り、Webサーバ220は受取った状況表示選択画面の表示制御データをWebクライアント100へ送り、Webクライアント100は受取った状況表示選択画面の表示制御データに基づいて画面を表示する。
- 20

- この画面でユーザが論理CORE作成を選択すると、今度は状況表示インタフェースプログラム224が第25図に示す論理CORE生成状況画面の論理COREの名称表示がない表示制御データを生成してWebサーバ220へ送り、Webサーバ220は受取った論理CORE生成状況画面の表示制御データをWeb
- 25

bクライアント100へ送り、Webクライアント100は受取った論理CORE生成状況画面の表示制御データに基づいて画面を表示する。

この画面でユーザがProject名を入力し表示ボタンを選択すると、Project名をWebクライアント100がWebサーバ220に送り、Webサーバ220からProject名を受取った状況表示インタフェースプログラム224は、指定のProject名のディレクトリから第14図のディレクトリASIC50にある論理COREのディレクトリ名称を抽出し、論理CORE生成状況画面の表示制御データを更新してWebサーバ220へ送り、Webサーバ220は受取った論理CORE生成状況画面の表示制御データをWebクライアント100へ送り、Webクライアント100は受取った論理CORE生成状況画面の表示制御データに基づいて画面を表示する。この画面には先に生成されたASICの論理COREのディレクトリ名称が表示される。

ここで、ASICの論理CORE生成状況表示画面での処理手順について説明する。第26図は、ASICの論理CORE生成状況表示画面での処理手順を示すフローチャートである。同図に示すように、ユーザが、表示された論理COREのどれかを選びダウンロードを選択すると（ステップS1501）、対象の論理COREをWebクライアント100にダウンロードすることができる（ステップS1502）。このダウンロードした論理COREは、ASICのチップレベルのネットリストとして論理検証に使用することができ、先に述べたようにインスタンスを構成する機能に絞ったデバッグが可能となる。

一方、ユーザが論理CORE名称を指定してFPGA化ボタンを選択すると（ステップS1503）、Webクライアント100は、Project名と論理CORE名称とFPGA化が選択されたこととをWebサーバ220に送り、Webサーバ220は受取ったデータを状況表示インタフェースプログラム224へ送る。状況表示インタフェースプログラム224は、第14図のディレクトリASIC50内にある、対象とする論理CORE名称のディレクトリ下のIOディレクトリ内にあるテーブルデータのファイル名を抽出し、その結果、第27図

に示すようなFPGA論理CORE生成インタフェース画面の表示制御データを
作成しWebサーバ220へ送る。

Webサーバ220は受取ったFPGA論理CORE生成インタフェース画面
の表示制御データをWebクライアント100へ送り、Webクライアント10
5 0は受取ったFPGA論理CORE生成インタフェース画面の表示制御データに
基づいて画面を表示する（ステップS1504）。この画面では、指定されたP
r o j e c t名と、論理CORE名と、論理COREを構成するインスタンス名
のリストが左側のリストボックスにリストされる。また、この画面では、対象の
論理COREを変えることも可能であり、ユーザがP r o j e c t名と論理CO
10 RE名を指定すると第25図で指定した同様の処理によって画面が更新される。

さらに、この画面は、対象のASICの論理COREを構成するインスタ
ンスのテーブルデータからFPGAの論理COREを生成するインタフェースであっ
て、ユーザは、左のリストボックスからインスタンスを指定し追加ボタンを選択
することで右のリストボックスにインスタンスを追加する。この右のリストボッ
15 クスにリストされたインスタンスが1つのFPGAとなる。ユーザがインスタン
スを選択し、FPGA名と管理のための追い番号である1以上の一意の整数であ
るFPGA番号を設定して実行ボタンを選択すると、Webクライアント100
が、対象のP r o j e c t名と、ASICの論理CORE名と、FPGA名と、
FPGAとするインスタンス名のリストと、FPGAの番号とをWebサーバ2
20 20へ送り、Webサーバ220は、受取ったデータを状況表示インタフェース
プログラム224へ送る。

そして、状況表示インタフェースプログラム224は、受取ったインスタ
ンス名のリストに従って第14図のディレクトリFPGA49に指定されたFPGA
の番号でディレクトリを作成し、第14図のディレクトリASIC50にある、
25 CORE名称のディレクトリ下のIOディレクトリから名称の一致するテーブル
データを前記番号のディレクトリへ複写し、テーブルデータ名のリストを引数と
してFPGA論理CORE生成プログラム252を実行する。

ここで、FPGA論理CORE生成プログラム252の処理手順について説明する。第28図及び第29図は、FPGA論理CORE生成プログラム252の処理手順を示すフローチャートである。第28図は第21図に示した処理と同じであり、また、第29図は第22図に示した処理とステップS1717及びステップS1720を除いて同じであるので、この2つのステップについて説明する。

第22図のステップS1017では、対象のインスタンスの入力であるポートに定義してある接続相手のインスタンス名とその出力ポート名が、メモリに格納している情報にない場合、エラーとしてログファイルに出力しているが、第29図のステップS1717では、元々ASICの論理COREとしてインスタンスのポート間接続は保証できているので、このステップS1717ではエラーではなくパッケージの入力端子情報としてメモリに追加格納する。

また、第22図のステップS1020では、最終的に残った未接続のポート情報をログファイルに出力している。一方、ステップS1720では、第14図のディレクトリASIC50下の”CORE+プロセスID+日付”で作成している対象とするASICの論理COREのディレクトリにあるログファイルとステップS1720で出力したログファイルとを比較し、ステップS1020で出力したログファイルにない出力ポートをパッケージの出力端子情報としてメモリに追加格納する。後は、ステップS1020の処理と同様にして第14図のディレクトリFPGA51の下に指定された番号のディレクトリ内に指定されたFPGA名称で論理COREをファイル出力する。

そして、状況表示インタフェースプログラム224は、FPGAの論理COREファイルを読み込み、パッケージの入出力となる端子数をカウントし、第30図に示す互換パッケージのテーブルの第2カラムにあるIO数に占める割合を全て計算し、割合が最大のパッケージとその互換パッケージデータを抽出し、第27図に示したFPGA論理CORE生成インタフェース画面のパッケージ名、IO使用率の画面表示制御データを更新し、Webサーバ220へ送る。Webサー

バ２２０は受取ったＦＰＧＡ論理ＣＯＲＥ生成インタフェース画面の表示制御データをＷｅｂクライアント１００へ送り、Ｗｅｂクライアント１００は受取ったＦＰＧＡ論理ＣＯＲＥ生成インタフェース画面の表示制御データに基づいて入力画面を更新する。

- 5 ユーザは、この結果に満足できなければ再度ＦＰＧＡ論理ＣＯＲＥ生成をやり直す。ユーザが決定を選択すると、Ｗｅｂクライアント１００は決定が選択されたことと、ＦＰＧＡ番号とパッケージ名とＩＯ使用率のデータとをＷｅｂサーバ２２０に送り、Ｗｅｂサーバ２２０からこれらのデータを受取った状況表示インタフェースプログラム２２４は、第１４図のディレクトリＦＰＧＡ５５に指定された番号のディレクトリを作成し、例えばplistの名称で受取ったデータをファイル出力する。

- 次に、論理合成インタフェースプログラム２２２の処理手順について説明する。第３１図は、論理合成インタフェースプログラム２２２の処理手順を示すフローチャートである。同図に示すように、この論理合成インタフェースプログラム
- 15 ２２２は、第３２図に示すような論理合成インタフェース画面の表示制御データをＷｅｂサーバ２２０へ送り、Ｗｅｂサーバ２２０は受取った論理合成インタフェース画面の表示制御データをＷｅｂクライアント１００へ送り、Ｗｅｂクライアント１００は受取った論理合成インタフェース画面の表示制御データに基づいて画面を表示する（ステップＳ２００１）。

- 20 ユーザがＰｒｏｊｅｃｔ名、第２５図で表示されている対象の論理ＣＯＲＥ名、論理合成する対象の機能ブロック名、組込むＦＰＧＡの番号を指定し、かつ面積優先か速度優先かの指定と、Ｄｅｂｕｇ（ＦＰＧＡの論理合成を実行しない）かＦｉｘ（ＦＰＧＡの論理合成を実行する）かを指定し（ステップＳ２００２）、実行ボタンを選択すると（ステップＳ２００３）、Ｗｅｂクライアント１００
- 25 はデータ選択画面を表示する。

ユーザがこの画面に従って論理合成するＲＴＬソースを選択すると、これらのデータをＷｅｂクライアント１００はＷｅｂサーバ２２０に送り、Ｗｅｂサーバ

220からデータを受取った論理合成インタフェースプログラム222は、RTLソースを第14図のディレクトリASIC52に指定の機能ブロック名でディレクトリを作成し、RTLを格納する（ステップS2004）。そして、RTLソースを1ファイルずつ読み込み、RTLソース内の変更履歴、版数などを記載するヘッダー部分にユーザが定義している動作周波数データを抽出し、抽出した値に対して20%増した周波数値とユーザに指定された速度優先または面積優先の論理合成条件を論理合成の制約として論理合成対象の機能ブロック名を指定してASICの論理合成ツールのASIC論理合成プログラム253を実行する（ステップS2005）。

- 10 ASICの論理合成が終了すると、論理合成インタフェースプログラム222は、第14図のディレクトリASIC54下にある指定の機能ブロックのディレクトリにASIC論理合成プログラム253が出力した結果のレポートファイルと、ユーザ管理サーバ240にProject名で問い合わせ得たユーザのメールアドレスとブロック名とをメールサーバ260に送り、メールサーバ260からユーザにメールを送信する（ステップS2006）。

そして、論理合成インタフェースプログラム222は、この処理過程を第14図のディレクトリASIC52下に指定の機能ブロック名で作成したディレクトリに存在する全RTLソースに対して実行する。全RTLソースの論理合成が終了すると（ステップS2007の肯定）、論理合成インタフェースプログラム222は、速度優先指定の場合は（ステップS2008の肯定）、全RTLの論理合成結果のレポートを検索し、RTLソースに定義された動作周波数を満足しているか否かを判断する（ステップS2009～ステップS2010）。面積重視の場合は何もしない。

- 次に、Fix指定を調べ（ステップS2011）、Fix指定されていれば、論理合成インタフェースプログラム222は、指定されたFPGA番号で第14図のディレクトリFPGA55下にディレクトリを作成し、この中に第14図のディレクトリASIC52下にある指定された機能ブロックディレクトリ内のR

TLを複写して格納し、指定されたProject名とFPGA番号とを引数としてFPGA論理合成インタフェースプログラムを実行する（ステップS2012）。

そして、論理合成インタフェースプログラム222は、論理合成ツールを起動し、第14図のディレクトリASIC50にある指定の論理COREのディレクトリからASICの論理COREを論理合成ツールにコマンドを投入して読み込ませ、ユーザからアップロードされ第14図のディレクトリASIC54に存在するASICのパッケージ端子割付けとASICと接続される外部デバイスと電氣的にインタフェースするバッファとを定義したファイルを読み込み、定義に従って論理合成ツールにコマンドを投入することでASICのバッファとスキューなどテスト回路をASICの論理COREの端子とパッケージの端子と接続されるチップのパッド間に挿入し接続させ、そして、論理合成済みである第14図のディレクトリASIC52下の全ディレクトリにある機能ブロックのデータを論理合成ツールにコマンドを投入して読み込んで論理COREに詰め込ませ、レイアウト設計用にASICのネットリストを生成させ、第14図のディレクトリASIC54にProject名でファイル出力させる（ステップS2013）。

次に、FPGA論理合成インタフェースプログラムの処理手順について説明する。第33図は、FPGA論理合成インタフェースプログラムの処理手順を示すフローチャートである。同図に示すように、このFPGA論理合成インタフェースプログラムは、ASICの論理合成インタフェースプログラムから指定されたFPGA番号で第14図のディレクトリFPGA55にディレクトリを作成し、第14図のディレクトリFPGA53下にある指定された番号のディレクトリ内のRTLソースを1ファイルずつ読み込み、RTLソース内の変更履歴、版数などを記載するヘッダー部分にユーザが定義している動作周波数データを抽出し、抽出した値に対して20%増した周波数値を論理合成の制約として論理合成対象の機能ブロック名を指定してFPGAの論理合成ツールのFPGA論理合成プログラム254を実行する（ステップS2201）。

FPGAの論理合成が終了すると、FPGA論理合成プログラム254は、第14図のディレクトリFPGA55にある対象とする番号のディレクトリに論理合成結果を機能ブロックの名称で出力する。こうすることでユーザは、後述する第34図の論理合成状況表示でゲート使用率を確認しつつ設計作業を進めることができる。そして、FPGA論理合成インタフェースプログラムは、論理合成の終了メッセージと、FPGAの番号と、ブロック名と、Project名でユーザ管理サーバ240に問い合わせ得たユーザのメールアドレスとをメールサーバ260に送り、メールサーバ260からユーザにメールを送信する（ステップS2202）。

次に、FPGA論理合成インタフェースプログラムは、指定された番号のFPGAを構成する機能ブロックの論理合成がすべて終了しているか確認する（ステップS2203）。この確認は、第14図のディレクトリFPGA51の対象とする番号のディレクトリにある機能ブロックの名称を抽出し、前記論理合成が終了した機能ブロック名と比較することで行なう。差分がなければ、次にFPGA論理合成インタフェースプログラムは、今度は、第14図のディレクトリFPGA51にある対象の番号のディレクトリにあるFPGA名である論理COREのファイル名を抽出し、この名称で第14図のディレクトリFPGA55にFPGAのレイアウトである、フィッティングに使用するFPGAの論理COREのポート名と同一であるパッケージの端子名と端子番号の対応、動作周波数、などを定義したフィッティング制御ファイルが、ユーザからアップロードされ存在しているかチェックする（ステップS2206）。

存在していれば、FPGA論理合成インタフェースプログラムは、FPGAを構成する全機能ブロックと第14図のディレクトリFPGA51下の、対象の番号のディレクトリにあるFPGAの論理COREとを指定し、FPGA論理合成プログラム254を実行する。そしてFPGAの論理合成が終了すると、FPGA論理合成インタフェースプログラムは、フィッティングプログラム255を起動し、論理合成結果のネットリストをフィッティングプログラム255に投入し

てレイアウトを行ない、FPGAのレイアウト結果であり回路情報を記録したROMデータを生成し、第14図のディレクトリROM46下の対象とする番号のディレクトリにFPGA名で出力させる（ステップS2207）。そして、ROMデータ生成の終了メッセージと、FPGAの番号と、Project名と、ユーザ管理サーバ240に問い合わせ得たユーザのメールアドレスとをメールサーバ260へ送り、メールを送信する（ステップS2208）。

差分があれば、後述するフィッティングデータの投入と実行画面で実行ボタンが選択されていないか判断し（ステップS2204）、選択されていなければ終了するが、選択されていれば論理合成が未の機能ブロック全てに対し、第14図のディレクトリFPGA51下にある対象の番号のディレクトリにある、FPGAの論理CORE内の各機能ブロック内部で、入出力ポート全てにフリップ・フロップを接続し、入力側の全フリップ・フロップの出力ポートを適当なゲート、例えば2入力NANDの入力ポートに接続し、全NANDゲートの出力を新たな2入力のNANDゲートの入力ポートに接続し、前述のように2入力NANDの多段接続を繰り返し、最終段の2入力NANDの出力を出力側の全フリップ・フロップの入力ポートに接続するHDL記述を挿入して論理合成可能な状態にする（ステップS2205）。

後は、差分がない場合と同一の処理となる。開発着手時点の検証戦略にも依るが、回路アーキテクチャ検討にて機能的に上りと下りに大きく分割できる機能ブロック構成となっている場合、上述のフリップ・フロップなどを用いたダミー回路を挿入することによって、上りの機能の設計が完了していれば、下りの機能の設計が未完であっても検証が可能であり、検証の効率化に貢献するところが大きい。

次に、第34図の論理合成状況表示画面について説明する。この画面は、ユーザがWebクライアント100で第24図の状況表示画面にある論理合成を選択すると、Webクライアント100は論理合成を選択されたことをWebサーバ220に送り、Webサーバ220からデータを受取った状況表示インターフェー

スプログラム 224 は、ログインしたユーザの名称でユーザ管理サーバ 240 に問い合わせ得た Project 名全てに対し、第 14 図のディレクトリ FPGA55 下の、全番号のディレクトリにある論理合成結果からゲート規模を抽出する。

- 5 そして、先に説明した FPGA 論理 CORE 生成で作成された同一ディレクトリにある plist からパッケージ情報を抽出し、このパッケージ情報を基に第 30 図のテーブル内でゲート規模の割合が 75% 以下になるパッケージを選び出し、第 34 図の論理合成状況表示画面の表示制御データを生成し、Web サーバ 220 に送る。Web サーバ 220 は、論理合成状況表示画面の表示制御データを Web クライアント 100 に送り、Web クライアント 100 は、受取った論理合成状況表示画面の表示制御データに基づいて画面を表示する。

- 10 ここで、ゲート規模の割合 75% について簡単に説明しておく。通常 FPGA の論理合成ツールは、論理合成の結果のゲート規模を基に論理合成ツールがもつデータベースから適当なパッケージとゲート使用率を示唆してくれる。しかしながら、FPGA のフィッティングにおいては、セルの配置過程において配線の効率を上げるため、FPGA で論理を実現する単位ブロックを配線に使用する。従って、論理合成結果よりゲート規模が大きくなることを考慮し、フィッティング後ゲート使用率が 100% を超えないように 75% に設定しているものであり、FPGA で異なることもある。このためにも先に説明した FPGA の論理合成を
- 15 機能ブロック毎に実行させることは有効である。

- 20 次に、フィッティングデータの投入と実行画面について説明する。第 35 図は、フィッティングデータの投入と実行画面の一例を示す図である。ユーザが Web クライアント 100 で第 18 図に示した手順画面にあるフィッティングを選択すると、Web クライアント 100 はフィッティングが選択されたことを Web
- 25 サーバ 220 に送り、Web サーバ 220 からデータを受取ったフィッティングインタフェースプログラム 223 は、ログインしたユーザの名称でユーザ管理サーバ 240 に問い合わせ得た Project 名全てに対し、第 14 図のディレ

クトリFPGA55下の、全ディレクトリの番号名と、それぞれのディレクトリにある論理合成結果であるネットリストのファイル名即ちFPGA名を抽出し、第35図に示すフィッティングデータの投入と実行画面の表示制御データを生成し、Webサーバ220に送る。

5 Webサーバ220はフィッティングデータの投入と実行画面の表示制御データをWebクライアント100に送り、Webクライアント100は、受取ったフィッティングデータの投入と実行画面の表示制御データに基づいて画面を表示する。この画面の条件データ投入を選択すると、Webクライアント100によりデータ選択画面が表示される。

10 この画面に従い、ユーザが先に述べたフィッティング制御ファイルを選択すると、Webクライアント100は、選択したFPGAの番号とフィッティング制御ファイルをWebサーバ220に送り、Webサーバ220からデータを受取ったフィッティングインタフェースプログラム223は、第14図のディレクトリFPGA55下の、対象の番号のディレクトリにフィッティング制御ファイルを格納する。

15 また、ユーザが実行ボタンを選択すると、Webクライアント100は、選択したFPGAの番号と、Project名と、実行が選択されたことをWebサーバ220に送り、Webサーバ220からデータを受取ったフィッティングインタフェースプログラム223は、Project名とFPGA番号とを引数としてFPGA論理合成インタフェースプログラムを実行する。FPGA論理合成
20 インタフェースプログラムの処理は先に説明した通りである。

次に、ROMデータ生成状況画面について説明する。第36図は、ROMデータ生成状況画面の一例を示す図である。ユーザがWebクライアント100で第24図に示した状況表示選択画面にあるROMデータを選択すると、Webクライアント100はROMデータを選択されたことをWebサーバ220に送る。

25 Webサーバ220からデータを受取った状況表示インタフェースプログラム224は、ログインしたユーザの名称でユーザ管理サーバ240に問い合わせ

得たProject名全てに対し、第14図のディレクトリROM46下の、全ディレクトリの番号名と、それぞれのディレクトリにあるフィッティング結果であるROMデータ名即ちFPGA名を抽出し、第36図に示したROMデータ生成状況画面の表示制御データを生成し、Webサーバ220に送る。

5 Webサーバ220は、ROMデータ生成状況画面の表示制御データをWebクライアント100に送り、Webクライアント100は、受取ったROMデータ生成状況画面の表示制御データに基づいて画面を表示する。ユーザは、この画面で対象とするFPGAの日付部分を選択すると、ROMデータをダウンロードすることができる。

10 次に、監視サーバ290の処理について説明する。まず、監視サーバ290が監視するものは、RTLの変更有無と、論理合成、フィッティング、レイアウト設計ツールの処理に要した時間と、設計者が定義する機能ブロックの変更規模を基にインプリメント設計に使用するツールの処理時間以外にかかる、変更反映に必要な手作業の時間である。

15 インプリメント設計者が管理する第14図のディレクトリLAYOUT47には、Project毎にProject名のディレクトリがあり、その中で作業を進める。各Project名のディレクトリにはレイアウト設計データが格納されるLAYディレクトリとRTLソースを格納するRTLディレクトリが存在し、RTLディレクトリ以下は、第14図のディレクトリASIC52以下と同一のディレクトリ構成となっている。

20 監視サーバ290は、LAYとRTLディレクトリ内を監視する。時間を集計した結果を反映するスケジュールは、Project名のディレクトリにScheduleの名称で格納されており、初期状態では、契約に基づいて、1stRTLの予定日とSign Offの予定日が設定されている。1stRTLとは、
25 、レイアウト設計が着手できる、その時点で大きな変更の見込みがなく、機能検証が80%程度終了しているRTLであると定義する。Sign Offとは、ASICのレイアウト設計完了後、ASIC製造データをデバイスベンダに渡す

日である。

第37図は、予定と実績画面の一例を示す図である。同図は、Schedule
e ファイルを画面表示制御データに変換し、Web クライアント100によって
表示される。すなわち、ユーザが第24図の状況表示画面で実績と予定を選択す
5 ると、Web クライアント100は、実績と予定が選択されたことをWeb サー
バ220経由で状況表示インタフェースプログラム224に送る。実績と予定が
選択されたことを受け取った状況表示インタフェースプログラム224は、S
chedule ファイルを画面表示制御データに変換し、Web サーバ220経由
でWeb クライアント100に送る。画面表示制御データを受け取ったWeb ク
10 ライアント100は、画面表示制御データに基づいて画面表示する。

また、第38図は、レイアウト設計者が設定する作業時間の設定ファイルの一
例を示す図である。このファイルは、Schedule ファイルと同一ディレク
トリにManual の名称で格納される。

監視サーバ290は、監視時間になると、ユーザ管理サーバ240からユーザ
15 名とユーザが所属するProject 名とメールアドレスを獲得する。そして、
Schedule ファイルを読みこみ1st RTL の予定日を抽出する。この予
定日が監視の実行年月日より未来であれば何もせず終了する。予定日が監視の実
行年月日より以前であれば、以降の処理を実行する。

まず、インプリメント設計者が管理するLAYOUT ディレクトリ下の、Pr
20 o j e c t 名のディレクトリにあるRTL ディレクトリにRTL ソースがなければ
処理を中断する。RTL ソースがあれば、第14図のディレクトリASIC 5
2 下の、各ブロックのディレクトリにあるRTL ソース間で比較する。

差分があれば、差分のあるブロック名とProject 名とインプリメント設
計者のメールアドレスをメールサーバ260に送り、メールサーバ260からイ
25 ンプリメント設計者に送信する。インプリメント設計者は、このメールによって
変更の反映にかかる時間の見積り作業を行なう。見積りの結果、インプリメント
設計者は、必要があればSchedule ファイルの時間データを更新または追

加、削除する。

次に、監視サーバ290は、第14図のディレクトリASIC50下の、作成時間が一番新しいASICの論理COREからASICを構成するブロック名を抽出し、メモリに格納する。そして、第14図のディレクトリASIC54下の、各ブロックのディレクトリにある論理合成結果のレポートファイルから論理合成処理時間を抽出する。先にメモリに格納したブロック名に対するレポートファイルが存在しない、つまり未だ論理合成されていない場合は、存在するレポートファイルから抽出した時間の平均時間を適用し、合計時間を算出しメモリに格納する。

そして今度は、インプリメント設計者が管理する第14図のディレクトリLAYOUT47下の、Project名のディレクトリ内にあるレイアウト設計結果のうち、レイアウト設計工程の最終工程であるタイミング検証結果のファイルが存在すれば、レイアウト設計の全行程で使用するツールが出力した処理結果ファイルから処理時間を抽出し、先にメモリに格納した時間と合せて合計時間を算出する。タイミング検証結果のファイルが存在しない場合は、契約時点にレイアウト設計者が仮に見積り予め監視サーバに設定した時間を適用する。

次に、監視サーバ290は、この合計時間と先に説明したManualに設定されている時間の合計をとり、1日を12時間として日数を算出し、Scheduleファイルが必要であれば更新する。つまり、最初の状態では、Scheduleファイルには1stRTLの予定日しかなく、この予定日に対して合計時間を加算し、第37図に示したように反映開始日としてインプリメント設計への変更取り込みの予定日を追加する。

次の監視では、反映開始日が監視の実行年月日より未来であれば、反映開始日の一つ前である1stRTL予定日から反映開始日として設定した予定日までの日数を比較対照とし、算出したインプリメント設計にかかる日数より長ければ反映開始日は更新しない。反映開始日が監視の実行年月日以前の時、監視サーバ290は、ユーザとインプリメント設計者のメールアドレスと反映開始の機会であ

るメッセージとをメールサーバ260へ送り、メールサーバ260からユーザとインプリメント設計者にメールを送信する。この時監視サーバ290は、一番新しい反映開始日が監視実行の年月日以前である時、反映開始日の実績日に年月日がなければScheduleファイルの更新はしない。

- 5 反映開始の機会であるメッセージのメールに対してユーザ及びインプリメント設計者双方から承認または拒絶のメールが監視サーバ290に届いた時、双方共承認であれば直ちにScheduleファイルの反映開始日の実績日にメールを受けた年月日を設定し、次の監視では、一番新しい反映予定日が監視実施の年月日以前であり、反映予定日の実績日に年月日があれば、実績日を基にインプリメント設計の日数を算出し、新たに反映開始日を追加する。この処理を続けて行くと反映開始日がSign Off日に近づく。

- 10 反映開始日の予定日を算出し、Sign Off日を超えた場合、監視サーバ290は、反映開始の予定日の設定が不可能な年月日にきているメッセージとユーザとインプリメント設計者のメールアドレスをメールサーバ260へ送り、メールサーバ260から双方へメールを送信する。

15 反映開始の機会であるメッセージのメールに対してユーザが拒絶している場合、ユーザに対しWebクライアント100から予定日を設定する旨のメッセージとユーザとインプリメント設計者のメールアドレスをメールサーバ260へ送り、メールサーバ260から双方へメールを送信する。

- 20 次に、予定と実績画面について説明する。第37図は、予定と実績画面の一例を示す図である。同図の論理Fixボタンと変更ボタンは、監視サーバ290の監視の実行年月日以前であり、一番新しい反映開始日のよこに配置される。ユーザがこのボタンの下にある年月日の設定欄に年月日を指定し変更ボタンを選択すると、Webクライアント100は、設定された年月日と変更が選択されたことをWebサーバ220に送り、Webサーバ220からデータを受取った状況表示
25 インタフェースプログラム224は、Scheduleファイル内の一番新しい反映開始日の予定日を指定された年月日に変更し、変更された旨のメッセージと

ユーザとインプリメント設計者のメールアドレスをメールサーバ260へ送り、メールサーバ260から双方へメールを送信する。

第37図内の論理Fixボタンが年月日の指定とともに選択された場合は、状況表示インタフェースプログラム224は、変更ボタンを選択された場合と同一の処理を実行後、監視サーバ290の監視を停止する処理を行なう。こうする事でユーザとインプリメント設計者双方の短期間の目標が明確となることと、インプリメント設計の試行回数が多いほど変更開始日の予定日の確度が上がり、これに対してユーザ及びインプリメント設計者双方共に進め方を考える契機を与える事が可能となり、結果として効率の向上に大きく貢献する。

上述してきたように、本実施の形態2では、Webクライアント100からのユーザの要求に基づいてASIC論理合成プログラム253がASICの論理合成を行ない、FPGA論理合成プログラム254がFPGAの論理合成を行ない、論理合成インタフェースプログラム222がWebクライアント100にASIC及びFPGAの論理合成の結果を表示し、メールサーバ260がASIC及びFPGAの論理合成の開始と結果をユーザに電子メールで通知することとしたので、ユーザは、論理合成の専任者を設置する必要がなく、いつでも論理合成が実行でき、専任者が実行したが如く論理合成品質を均一に保つことができるとともに、電子メールで論理合成の開始と結果の通知を受けることができ、論理合成の進捗を定期的にコンピュータで確認する必要がなくなる。

また、本実施の形態2では、Webクライアント100からのユーザの要求に基づいて、ASIC論理CORE生成プログラム251がASICを構成する機能ブロックからユーザが指定する複数の機能ブロックのポート間接続情報のみから成るネットリストを生成し、FPGA論理CORE生成プログラム252が生成したネットリストに論理合成済み対象機能ブロックのデータを埋め込んでプログラマブル・ロジックデバイスの回路を記録したROMデータを生成し、論理CORE生成インタフェースプログラム221が生成したROMデータの生成結果をコンピュータに表示するとともにメールサーバ260がユーザに電子メールで

通知する構成としたので、ユーザは、プログラマブル・ロジックデバイス専用の開発環境が必要なく、プログラマブル・ロジックデバイスの回路を記録したROMデータを作成する負荷と、時間と、かかるコストとを削減することができる。

また、本実施の形態2では、ユーザが指定したASICを構成する機能ブロックの設計が未完了で回路データが存在しない場合に、FPGA論理CORE生成プログラム252が当該機能ブロックの入力端子及び出力端子に仮のフリップ・フロップなどを用いた回路を挿入したネットリストを生成することとしたので、プログラマブル・ロジックデバイスのプロトタイピングによる検証において、検証対象ではない機能ブロックの設計が完了していなくとも、プロトタイピングによる検証を進めることができ、検証の効率化を図ることができる。

また、本実施の形態2では、監視サーバ290が、ユーザが持つ最新の回路データとインプリメント設計者がインプリメント設計に取り込んでいる回路データ間の変更の規模を監視し、監視結果及びレイアウト設計にかかる時間に基づいて計画された日時に到達すると、メールサーバ260が変更をASICのインプリメント設計へ反映するタイミングにあることをユーザ及びASICのインプリメント設計者に電子メールで通知し、この通知に応答してユーザが反映の日付を変更することで停止を要求することとしたので、発生した変更を効率よくASICのレイアウト設計に反映することができるとともに、変更を反映するタイミングを設定することで、ユーザはいつまで変更することが可能かを判断することができ、早い段階でスケジュールの見直しができる。

以上説明したように、本発明によれば以下の効果が得られる。

請求の範囲第1項記載の発明によれば、請求の範囲第1項の発明のコア（論理コア）を生成する手法をコンピュータに実行させるプログラムを記憶したプログラム記憶媒体によれば、集積回路開発において、RTL設計の入力となるブロックのポート仕様の品質を事前に確保できる効果があり、更に機能ブロックが多く存在し、設計リソースも多い大規模集積回路の開発においても、事前にブロック間の接続が確認できていることにより、必ずチップが組み上がることを保証する

ことができる。

請求の範囲第2項記載の発明によれば、ASICのコア（論理コア）からプログラマブル・ロジックデバイスのコア（論理コア）としてブロック間接続を保持したまま切り出すため、本コア（論理コア）を用いたプログラマブル・ロジック
5 デバイスの検証を実施すれば、ASICにおいては、少なくとも同一の構成での検証の重複を回避することができる。よって、ASICとプログラマブル・ロジックデバイスのコンカレント開発を効率よく進めることができる。

また、I/Oバッファを挿入する際、チップの端子情報から仮のコア（論理コア）を生成し、ブロックから生成したコア（論理コア）と入れ替える手段によっ
10 て、ブロックに定義されているチップのポート情報とチップの端子情報をクロスチェックでき、ブロックのポート仕様とチップ端子仕様の両者の品質を確保することができる。

請求の範囲第3項記載の発明によれば、任意の規模、個数のブロックのポートとポートの接続情報からなるネットリストを生成することができる。

請求項4記載の発明によれば、任意の規模、個数のブロックのポートとポート
15 の接続情報からなるネットリストから、チップネットリストを生成することができる。

請求の範囲第5項記載の発明によれば、プログラマブル・ロジックデバイスのコア（論理コア）を生成する手法を論理合成ツールに実行させるプログラムを記憶した記憶媒体によれば、ポートの機能を判断することが難しいネット名がポ
20 ートの名称となる場合の論理検証等デバッグの効率の低下を防ぐことができる。

上述のようにして、設計ドキュメントのデータからコア（論理コア）を生成し、該コア（論理コア）から階層構造及び接続情報を保持した状態でプログラマブル・ロジックデバイス用に新たにコア（論理コア）を生成することで、回路アー
25 キテクチャを共有することができ、回路データを埋め込み、機能検証したインスタンス内のデバイス・テクノロジーに依存しない回路データ及びインスタンス間のネットは、ASIC化した場合に再検証を回避することが可能となり、ASIC

とプログラマブル・ロジックデバイス間の違いによる再設計も回避することが可能となる。

このように、本発明によれば、アーキテクチャの共有化を実現し、再設計と再検証を極力回避することができる集積回路の開発方法及び集積回路の開発方法を
5 記憶したプログラマブル記憶媒体を提供することができる。

また、本発明によれば、ASICを構成する機能ブロックをポート間接続情報に基づいてグループ化し、グループ化した機能ブロックのポートとポート間接続情報からなるネットリストをプログラマブル・ロジックデバイスのコア（論理コア）として生成し、ASICを構成する機能ブロックの回路データからASIC
10 用論理合成データ及びプログラマブル・ロジックデバイス用論理合成データを作成し、グループ化した機能ブロックについてのプログラマブル・ロジックデバイス用論理合成データを生成したネットリストに埋め込んでプログラマブル・ロジックデバイスの回路を記録した実機評価用ROMデータを生成し、作成したASIC
15 用論理合成データを用いてASICのレイアウト作成およびタイミング検証を実機評価用ROMデータの生成とコンカレントに行ない、生成したROMデータを用いた実機評価結果に基づく回路データの変更をASICのレイアウト作成およびタイミング検証に反映することとしたので、ASICとプログラマブル・ロジックデバイスの効率的なコンカレント開発が可能になり、ASICの開発期間を短縮することができるという効果を奏する。

20 また、本発明によれば、ASICを構成する機能ブロックをポート間接続情報に基づいてグループ化し、グループ化した機能ブロックのポートとポート間接続情報からなるネットリストをプログラマブル・ロジックデバイスのコア（論理コア）として生成し、ASICを構成する機能ブロックの回路データからASIC
25 用論理合成データ及びプログラマブル・ロジックデバイス用論理合成データを作成し、グループ化した機能ブロックについてのプログラマブル・ロジックデバイス用論理合成データを生成したネットリストに埋め込んでプログラマブル・ロジックデバイスの回路を記録した実機評価用ROMデータを生成し、作成したAS

I C用論理合成データを用いてA S I Cのレイアウト作成およびタイミング検証を実機評価用R O Mデータの生成とコンカレントに行なうよう構成したので、A S I Cとプログラマブル・ロジックデバイスの効率的なコンカレント開発が可能になり、A S I Cの開発期間を短縮することができるという効果を奏する。

- 5 また、本発明によれば、ユーザの要求に応じてA S I Cの論理合成を実行し、作成したA S I Cの論理合成結果がユーザの要求する速度性能を満足したか否かを判断し、判断結果に基づいてプログラマブル・ロジックデバイスの論理合成を実行し、A S I C論理合成の実行結果及びプログラマブル・ロジックデバイス論理合成の実行結果をコンピュータに表示し、A S I C論理合成の実行開始と実行結果及びプログラマブル・ロジックデバイス論理合成の実行開始と実行結果をユーザに電子メールで通知する構成としたので、ユーザは、論理合成の専任者を設置する必要がなく、いつでも論理合成が実行でき、専任者が実行したが如く論理合成品質を均一に保つことができるとともに、電子メールで論理合成の開始と結果の通知を受けることができ、論理合成の進捗を定期的にコンピュータで確認する
10 必要がなくなるという効果を奏する。
15

- また、本発明によれば、ユーザの要求に応じてA S I Cを構成する機能ブロックからユーザが指定する複数の機能ブロックのポート間接続情報から成るネットリストを生成し、生成したネットリストに論理合成済み対象機能ブロックのデータを埋め込んでプログラマブル・ロジックデバイスの回路を記録したR O Mデータを生成し、生成したR O Mデータの生成結果をコンピュータに表示するとともにユーザに電子メールで通知する構成としたので、ユーザは、プログラマブル・ロジックデバイス専用の開発環境が必要なく、プログラマブル・ロジックデバイスの回路を記録したR O Mデータを作成する負荷と、時間と、かかるコストとを削減することができるという効果を奏する。
20

- 25 また、本発明によれば、ユーザが指定したA S I Cを構成する機能ブロックの設計が未完了で回路データが存在しない場合に、当該機能ブロックの入力端子及び出力端子に仮のフリップ・フロップなどを用いた回路を挿入したネットリスト

を生成することとしたので、プログラマブル・ロジックデバイスのプロトタイピングによる検証において、検証対象ではない機能ブロックの設計が完了していなくとも、プロトタイピングによる検証を進めることができ、検証の効率化を図ることができるという効果を奏する。

5 また、本発明によれば、ユーザが持つ最新の回路データとインプリメント設計者がインプリメント設計に取り込んでいる回路データ間の変更の規模を監視し、監視結果及びレイアウト設計にかかる時間に基づいて計画された日時に到達すると変更をA S I Cのインプリメント設計へ反映するタイミングにあることをユーザ及びA S I Cのインプリメント設計者に電子メールで通知し、この通知に
10 してユーザが反映の日付を変更することで停止を要求することとしたので、発生した変更を効率よくA S I Cのレイアウト設計に反映することができるとともに、変更を反映するタイミングを設定することでユーザはいつまで変更することが可能かを判断することができ、早い段階でスケジュールの見直しができるという効果を奏する。

15 また、本発明によれば、一方では、複数の機能ブロックの全部又は一部の機能ブロックを含むF P G Aの端子情報が記述された第1デザインと、第1デザインの下位層として記述されたF P G Aと同様の端子情報が記述された第2デザインとから、第1デザイン、第2デザインにおける同一端子間が接続され、かつ、端子間にF P G A対応のバッファが挿入されたF P G A用デザイン情報を生成し、
20 他方では、複数の機能ブロックを含むA S I Cの端子情報が記述された第3デザインと、第3デザインの下位層として記述されたA S I Cと同様の端子情報が記述された第4デザインとから、第3デザイン、第4デザイン間における同一端子間が接続され、かつ、端子間にA S I C対応のバッファが挿入されたA S I C用デザイン情報を生成し、第2デザイン、第4デザインのそれぞれを、各デザイン
25 が含む機能ブロックの接続情報に基づいて生成された回路情報で置き換える、ことでF P G A及びA S I Cのネットリストを生成することとしたので、A S I Cとプログラマブル・ロジックデバイスの効率的なコンカレント開発が可能になり

、A S I Cの開発期間を短縮することができるという効果を奏する。

産業上の利用可能性

5 以上のように、本発明に係る集積回路の開発方法及び集積回路の開発方法を記憶したプログラム記憶媒体、ならびにA S I Cとプログラマブル・ロジックデバイスのコンカレント開発システム、開発プログラム及び開発方法は、集積回路の開発、特にA S I Cとプログラマブル・ロジックデバイスの開発に適している。

請 求 の 範 囲

1. ブロックのポートとポートの接続情報からなるASICのコア（論理コア）から、接続関係のある任意のブロックを選択しグループ化する手段により任意の規模、個数のブロックのポートとポートの接続情報からなる、論理合成ツールが読み込み可能なHDLフォーマットのコア（論理コア）を生成し（ステップ1）、
- 5 チップの端子情報から論理合成ツールで仮のチップデザインを作成し、このデザインに端子を発生させ（ステップ2）、
- 10 作成したデザイン内部にステップ2と同一のデザインをセルとして発生させ（ステップ3）、
- デザインとセル間の同一名称のポートを接続し（ステップ4）、
- 接続したポート間のネットに対し、デバイス・テクノロジーに依存するI/Oバッファを挿入し（ステップ5）、
- 15 ステップ1で作成したコア（論理コア）とセルを入れ替え、トップ階層であるデザインの階層を展開し、ネットリストを生成する（ステップ6）、
- ことを特徴とする集積回路の開発方法を記憶したプログラム記憶媒体。
2. ASICとプログラマブル・ロジックデバイスのコンカレント開発を行なう場合において、
- 20 ブロックのポートとポートの接続情報からなるASICのコア（論理コア）と呼ぶネットリストから、接続関係のある任意のブロックを選択しグループ化する手段により、任意の規模、個数のブロックのポートとポートの接続情報からなるコア（論理コア）を生成し（ステップ1）、
- 25 チップの端子情報から論理合成ツールで仮のチップのデザインを作成し（ステップ2）、
- このデザインに前記チップの端子情報にある名称で端子を発生させ（ステップ

3)、

この内部に前記と同一デザインをセルとして発生させ（ステップ4）、

前記デザインとセル間の同一名称のポートを接続し（ステップ5）、

この接続ネットに対し、チップの端子情報からデバイス・テクノロジーに依存す

5 るI/Oバッファを挿入し（ステップ6）、

ステップ1で生成したコア（論理コア）を前記セルと入れ替え（ステップ7）

、

このコア（論理コア）のトップ階層である、作成したデザインの階層を展開し、
プログラマブル・ロジックデバイスのチップのネットリストを生成する（ステ

10 ュップ8）、

ことを特徴とする集積回路の開発方法。

3. ブロックのポートとポートの接続情報からなるASICのコア（論理コア）から、接続関係のある任意のブロックを選択しグループ化する手段により、任意の規模、個数のブロックのポートとポートの接続情報からなるネットリストを生成することを特徴とする請求の範囲第1項に記載のプログラム記憶媒体。

15

4. ブロックのポートとポートの接続情報からなるASICのコア（論理コア）から、接続関係のある任意のブロックを選択しグループ化する手段により、任意の規模、個数のブロックのポートとポートの接続情報からなるコア（論理コア）を生成し（ステップ1）、

20

チップの端子情報から、論理合成ツールで仮のチップのデザインを作成し、このデザインに前記チップの端子情報にある名称で端子を発生させ（ステップ2）

、

この内部に前記と同一デザインをセルとして発生させ（ステップ3）、

25

前記デザインとセル間の同一名称のポートを接続し（ステップ4）、

この接続ネットに対しチップの端子情報からデバイス・テクノロジーに依存する

I/Oバッファを挿入し（ステップ5）、

ステップ1で生成したコア（論理コア）を前記セルと入れ替え（ステップ6）

、

このコア（論理コア）のトップの階層であるステップ2で作成したデザインの

5 階層を展開してチップネットリストを生成する（ステップ7）、

ことを特徴とする集積回路の開発方法を記録したプログラム記憶媒体。

5. 任意の規模、個数のブロックのポートとポートの接続情報からなるネット
リストを生成する場合において、

10 トップ階層のポート名をネット名から接続先のブロックのポート名と一致する
ようにポート名を修正し、ブロックのポートとポートの接続情報からなるネット
リストを生成することを特徴とする請求の範囲第2項に記載の集積回路の開発方
法。

15 6. ASICを構成する機能ブロックをポート間接続情報に基づいてグループ
化し、該グループ化した機能ブロックのポートとポート間接続情報からなるネット
リストをプログラマブル・ロジックデバイスのコア（論理コア）として生成す
るネットリスト生成工程と、

前記ASICを構成する機能ブロックの回路データからASIC用論理合成デ
ータ及びプログラマブル・ロジックデバイス用論理合成データを作成する論理合
20 成工程と、

前記グループ化した機能ブロックについてのプログラマブル・ロジックデバイ
ス用論理合成データを前記ネットリスト生成工程により生成されたネットリスト
に埋め込んでプログラマブル・ロジックデバイスの回路を記録した実機評価用R

25 OMデータを生成するROMデータ生成工程と、

前記論理合成工程により作成されたASIC用論理合成データを用いてAS I
Cのレイアウト作成およびタイミング検証を前記ROMデータ生成工程による実

機評価用ROMデータの生成とコンカレントに行なうレイアウト作成工程と、

前記ROMデータ生成工程により生成されたROMデータを用いた実機評価結果に基づく回路データの変更を前記ASICのレイアウト作成およびタイミング検証に反映する差分反映工程と、

5 を含んだことを特徴とするASICとプログラマブル・ロジックデバイスのコンカレント開発方法。

7. 前記レイアウト作成工程は、前記ROMデータ生成工程より前か同時に始まり、前記差分反映工程は、前記レイアウト作成工程開始後におこなわれた前記
10 回路データの変更をASICのレイアウト作成およびタイミング検証に反映することを特徴とする請求の範囲第6項に記載のASICとプログラマブル・ロジックデバイスのコンカレント開発方法。

8. 前記ROMデータ生成工程は、前記グループ化した機能ブロックのうち前
15 記論理合成工程によりプログラマブル・ロジックデバイス用論理合成データが作成されていない機能ブロックに対して、入力端子及び出力端子にダミー回路を挿入したプログラマブル・ロジックデバイス用論理合成データを前記ネットリスト生成工程により生成されたネットリストに埋め込んでプログラマブル・ロジック
20 デバイスの回路を記録した実機評価用ROMデータを生成することを特徴とする請求の範囲第6項または第7項に記載のASICとプログラマブル・ロジックデバイスのコンカレント開発方法。

9. ASICを構成する機能ブロックをポート間接続情報に基づいてグループ化し、該グループ化した機能ブロックのポートとポート間接続情報からなるネット
25 リストをプログラマブル・ロジックデバイスのコア（論理コア）として生成するネットリスト生成手段と、

前記ASICを構成する機能ブロックの回路データからASIC用論理合成デ

ータ及びプログラマブル・ロジックデバイス用論理合成データを作成する論理合成手段と、

前記グループ化した機能ブロックについてのプログラマブル・ロジックデバイス用論理合成データを前記ネットリスト生成手段により生成されたネットリスト
5 に埋め込んでプログラマブル・ロジックデバイスの回路を記録した実機評価用ROMデータを生成するROMデータ生成手段と、

前記論理合成手段により作成されたASIC用論理合成データを用いてASICのレイアウト作成およびタイミング検証を前記ROMデータ生成手段による実機評価用ROMデータの生成とコンカレントに行なうレイアウト作成手段と、

10 を備えたことを特徴とするASICとプログラマブル・ロジックデバイスのコンカレント開発システム。

10. ASICを構成する機能ブロックをポート間接続情報に基づいてグループ化し、該グループ化した機能ブロックのポートとポート間接続情報からなるネットリストをプログラマブル・ロジックデバイスのコア（論理コア）として生成
15 するネットリスト生成手順と、

前記ASICを構成する機能ブロックの回路データからASIC用論理合成データ及びプログラマブル・ロジックデバイス用論理合成データを作成する論理合成手順と、

20 前記グループ化した機能ブロックについてのプログラマブル・ロジックデバイス用論理合成データを前記ネットリスト生成手順により生成されたネットリストに埋め込んでプログラマブル・ロジックデバイスの回路を記録した実機評価用ROMデータを生成するROMデータ生成手順と、

前記論理合成手順により作成されたASIC用論理合成データを用いてASIC
25 Cのレイアウト作成およびタイミング検証を前記ROMデータ生成手順による実機評価用ROMデータの生成とコンカレントに行なうレイアウト作成手順と、

前記ROMデータ生成手順により生成されたROMデータを用いた実機評価結

果に基づく回路データの変更を前記ASICのレイアウト作成およびタイミング検証に反映する差分反映手順と、

をコンピュータに実行させることを特徴とするASICとプログラマブル・ロジックデバイスのコンカレント開発プログラム。

5

11. ネットワークに接続されたコンピュータからユーザが利用するASICとプログラマブル・ロジックデバイスのコンカレント開発システムにおいて、

前記ユーザの要求に応じてASICの論理合成を実行するASIC論理合成手段と、

10 前記ASIC論理合成手段により作成されたASICの論理合成結果が前記ユーザの要求する速度性能を満足したか否かを判断するASIC論理合成結果判断手段と、

前記ASIC論理合成結果判断手段による判断結果に基づいてプログラマブル・ロジックデバイスの論理合成を実行するプログラマブル・ロジックデバイス論

15 理合成手段と、

前記ASIC論理合成手段によるASIC論理合成の実行結果及び前記プログラマブル・ロジックデバイス論理合成手段によるプログラマブル・ロジックデバイス論理合成の実行結果を前記コンピュータに表示する論理合成結果表示手段と、

20 前記ASIC論理合成手段によるASIC論理合成の実行開始と実行結果及び前記プログラマブル・ロジックデバイス論理合成手段によるプログラマブル・ロジックデバイス論理合成の実行開始と実行結果を前記ユーザに電子メールで通知する論理合成通知手段と、

を備えたことを特徴とするASICとプログラマブル・ロジックデバイスのコンカレント開発システム。

25

12. 前記ユーザの要求に応じて前記ASICを構成する機能ブロックから前

記ユーザが指定する複数の機能ブロックのポート間接続情報から成るネットリストを生成するネットリスト生成手段と、前記ネットリスト生成手段により生成されたネットリストに論理合成済み対象機能ブロックのデータを埋め込んでプログラマブル・ロジックデバイスの回路を記録したROMデータを生成するROMデータ生成手段と、前記ROMデータ生成手段により生成されたROMデータの生成結果を前記コンピュータに表示するROMデータ生成結果表示手段と、前記ROMデータ生成手段により生成されたROMデータの生成結果を前記ユーザに電子メールで通知するROMデータ生成結果通知手段とをさらに備えたことを特徴とする請求の範囲第11項に記載のASICとプログラマブル・ロジックデバイスのコンカレント開発システム。

13. 前記ユーザが指定した前記ASICを構成する機能ブロックの設計が未完了で回路データが存在しない場合に、当該機能ブロックの入力端子及び出力端子にダミー回路を挿入したネットリストを生成する仮ネットリスト生成手段をさらに備えたことを特徴とする請求の範囲第12項に記載のASICとプログラマブル・ロジックデバイスのコンカレント開発システム。

14. 前記ユーザが持つ最新の回路データとインプリメント設計者がインプリメント設計に取り込んでいる回路データ間の変更の規模を監視する監視手段と、前記監視手段による監視結果及びレイアウト設計にかかる時間に基づいて計画された日時に到達すると前記変更を前記ASICのインプリメント設計への反映のタイミングにあることを前記ユーザ及び前記ASICのインプリメント設計者に電子メールで通知する変更タイミング通知手段と、前記変更タイミング通知手段に応答して前記ユーザが前記反映の日付を変更することで停止を要求する反映停止要求手段とをさらに備えたことを特徴とする請求の範囲第11項、第12項または第13項に記載のASICとプログラマブル・ロジックデバイスのコンカレント開発システム。

15. ネットワークに接続されたコンピュータからユーザが利用するASICとプログラマブル・ロジックデバイスのコンカレント開発方法において、

5 前記ユーザの要求に応じてASICの論理合成を実行するASIC論理合成工程と、

前記ASIC論理合成工程により作成されたASICの論理合成結果が前記ユーザの要求する速度性能を満足したか否かを判断するASIC論理合成結果判断工程と、

10 前記ASIC論理合成結果判断工程による判断結果に基づいてプログラマブル・ロジックデバイスの論理合成を実行するプログラマブル・ロジックデバイス論理合成工程と、

15 前記ASIC論理合成工程によるASIC論理合成の実行結果及び前記プログラマブル・ロジックデバイス論理合成工程によるプログラマブル・ロジックデバイス論理合成の実行結果を前記コンピュータに表示する論理合成結果表示工程と、

前記ASIC論理合成工程によるASIC論理合成の実行開始と実行結果及び前記プログラマブル・ロジックデバイス論理合成工程によるプログラマブル・ロジックデバイス論理合成の実行開始と実行結果を前記ユーザに電子メールで通知する論理合成通知工程と、

20 を含んだことを特徴とするASICとプログラマブル・ロジックデバイスのコンカレント開発方法。

16. 一方では、複数の機能ブロックの全部又は一部の機能ブロックを含むFPGAの端子情報が記述された第1デザインと、該第1デザインの下位層として記述された該FPGAと同様の端子情報が記述された第2デザインとから、該第1デザイン、第2デザインにおける同一端子間が接続され、かつ、該端子間にFPGA対応のバッファが挿入されたFPGA用デザイン情報を生成し、

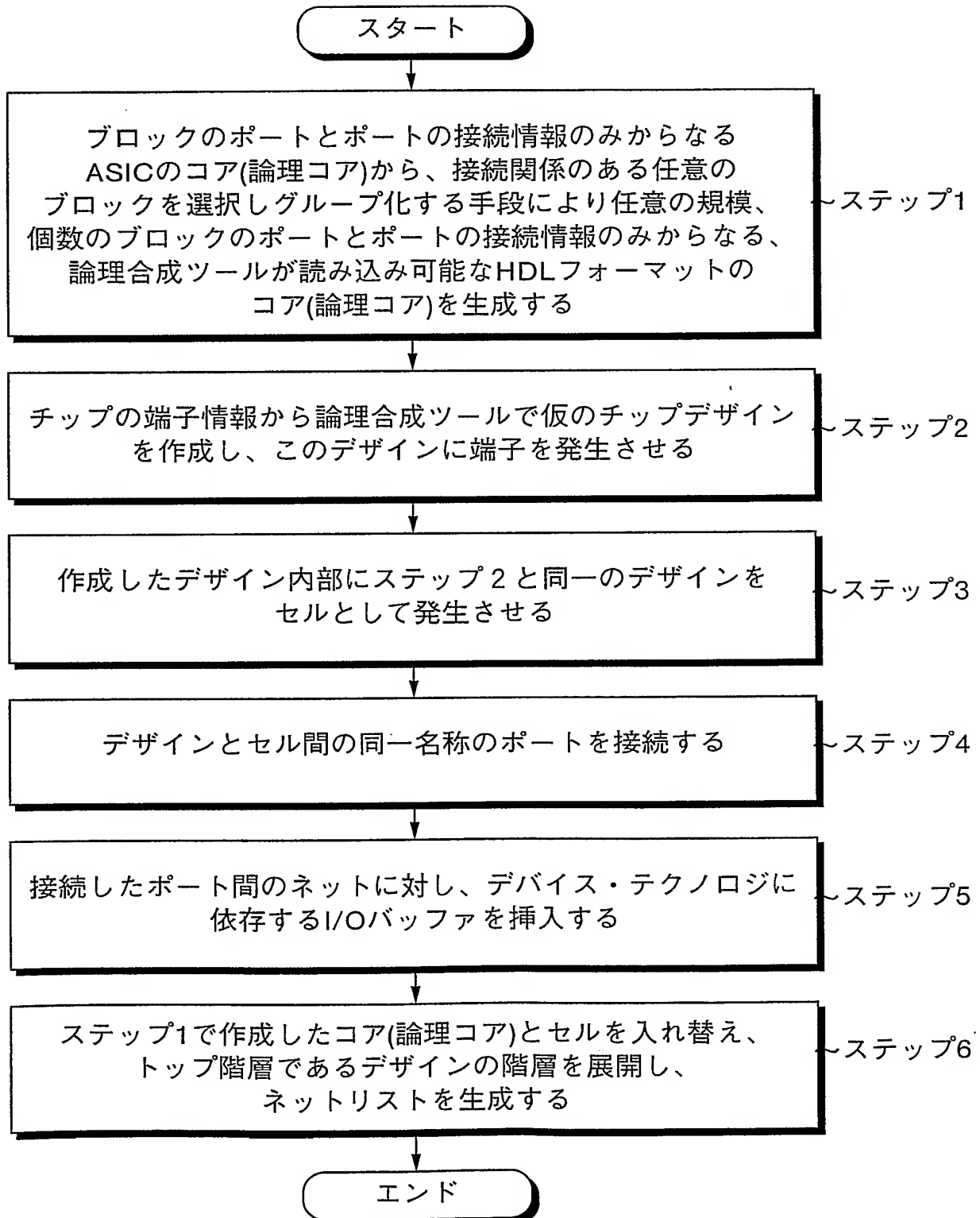
25

他方では、該複数の機能ブロックを含むA S I Cの端子情報が記述された第 3 デザインと、該第 3 デザインの下位層として記述された該A S I Cと同様の端子情報が記述された第 4 デザインとから、該第 3 デザイン、第 4 デザイン間における同一端子間が接続され、かつ、該端子間にA S I C対応のバッファが挿入されたA S I C用デザイン情報を生成し、

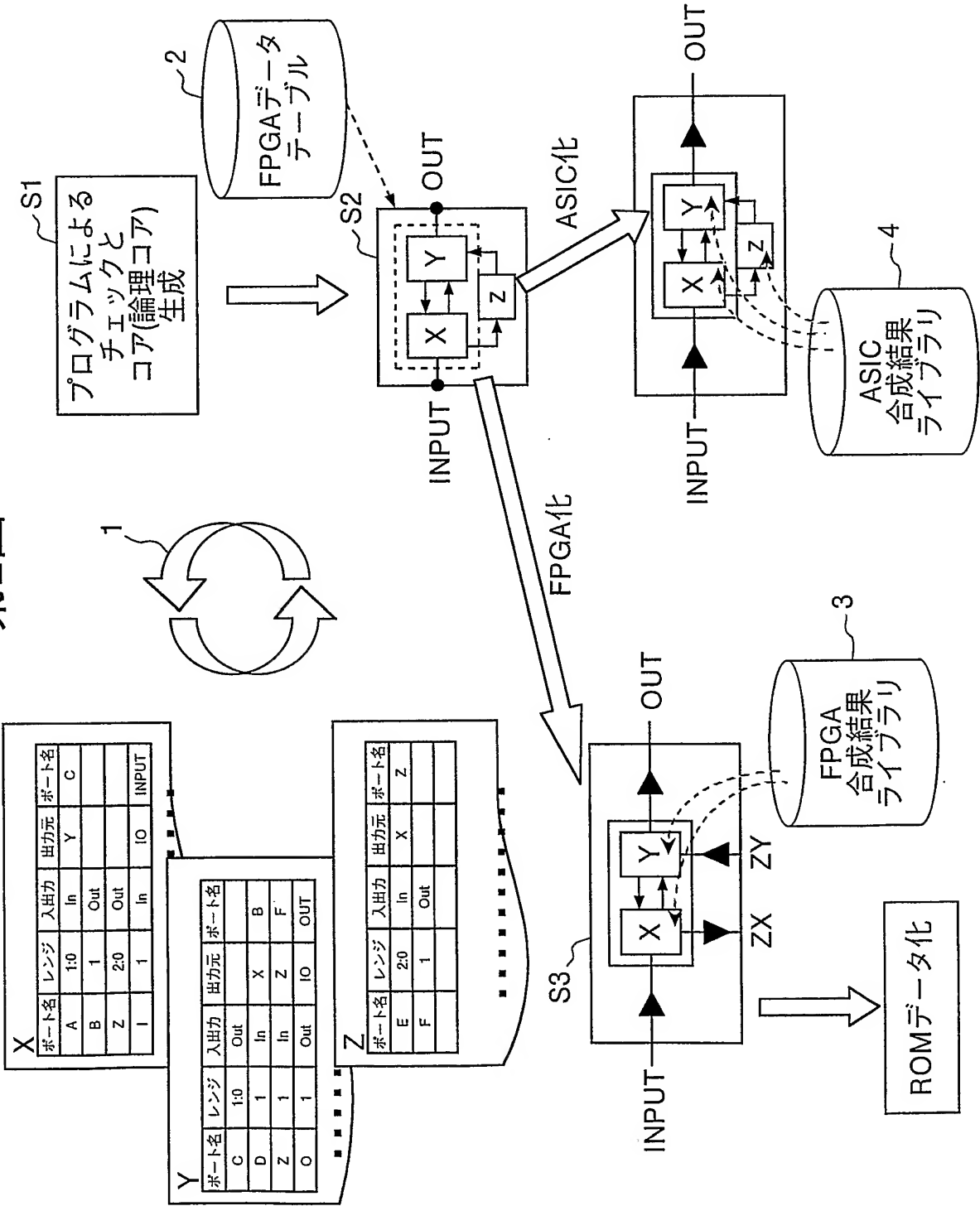
5

前記第 2 デザイン、第 4 デザインのそれぞれを、各デザインが含む機能ブロックの接続情報に基づいて生成された回路情報で置き換える、
ことでF P G A及びA S I Cのネットリストを生成する方法。

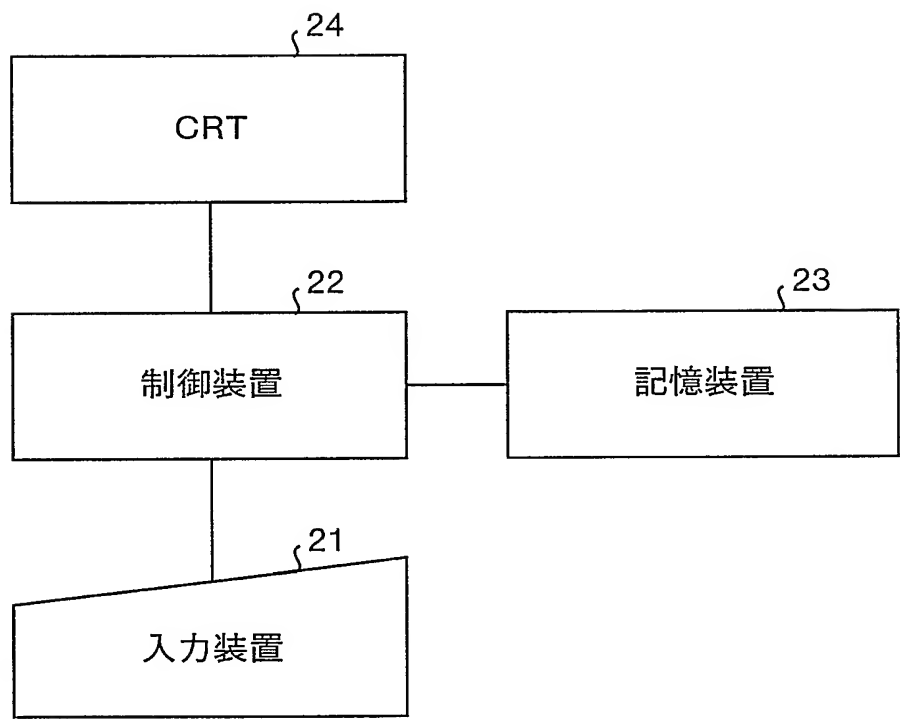
第1図



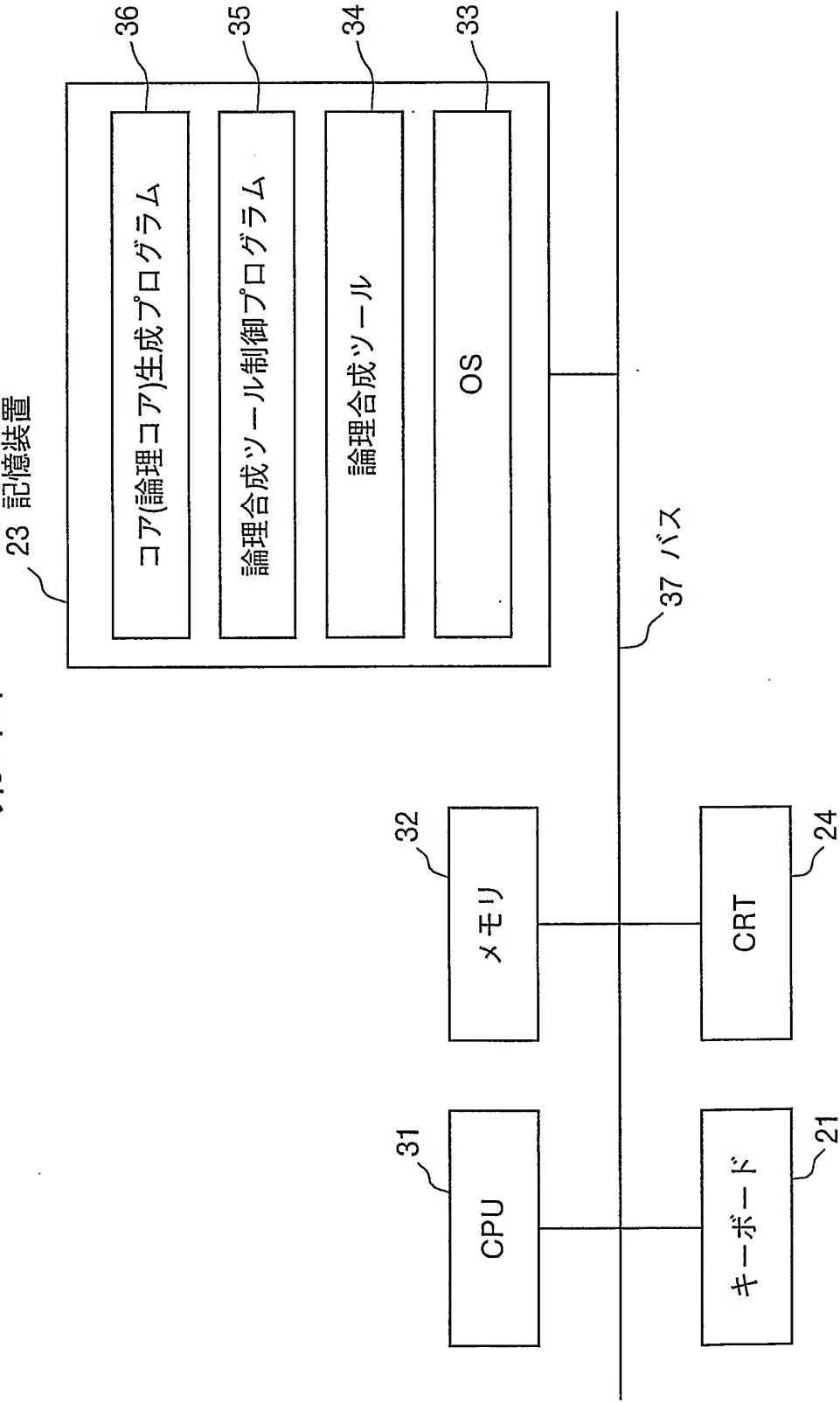
第2図



第3図

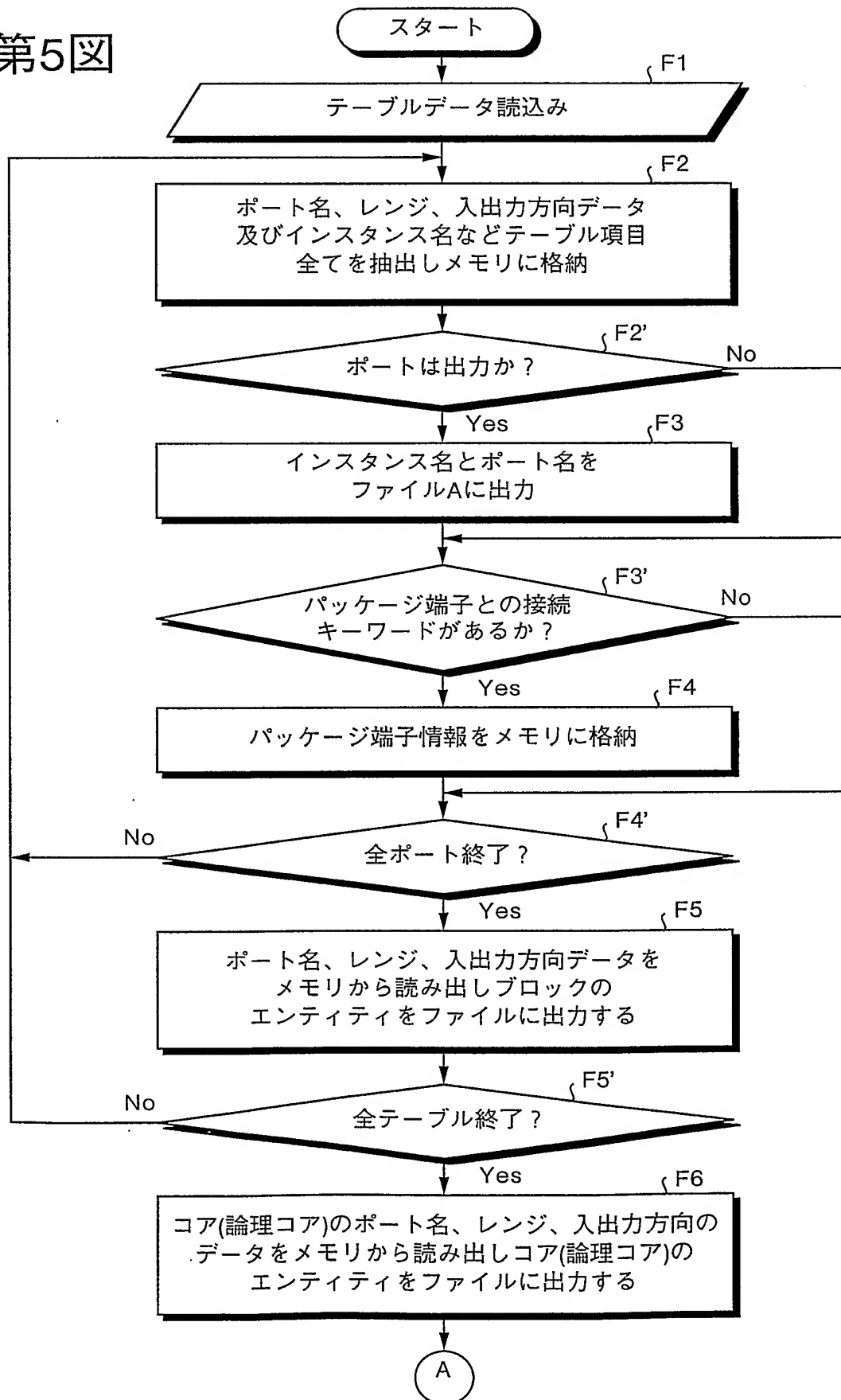


第4図



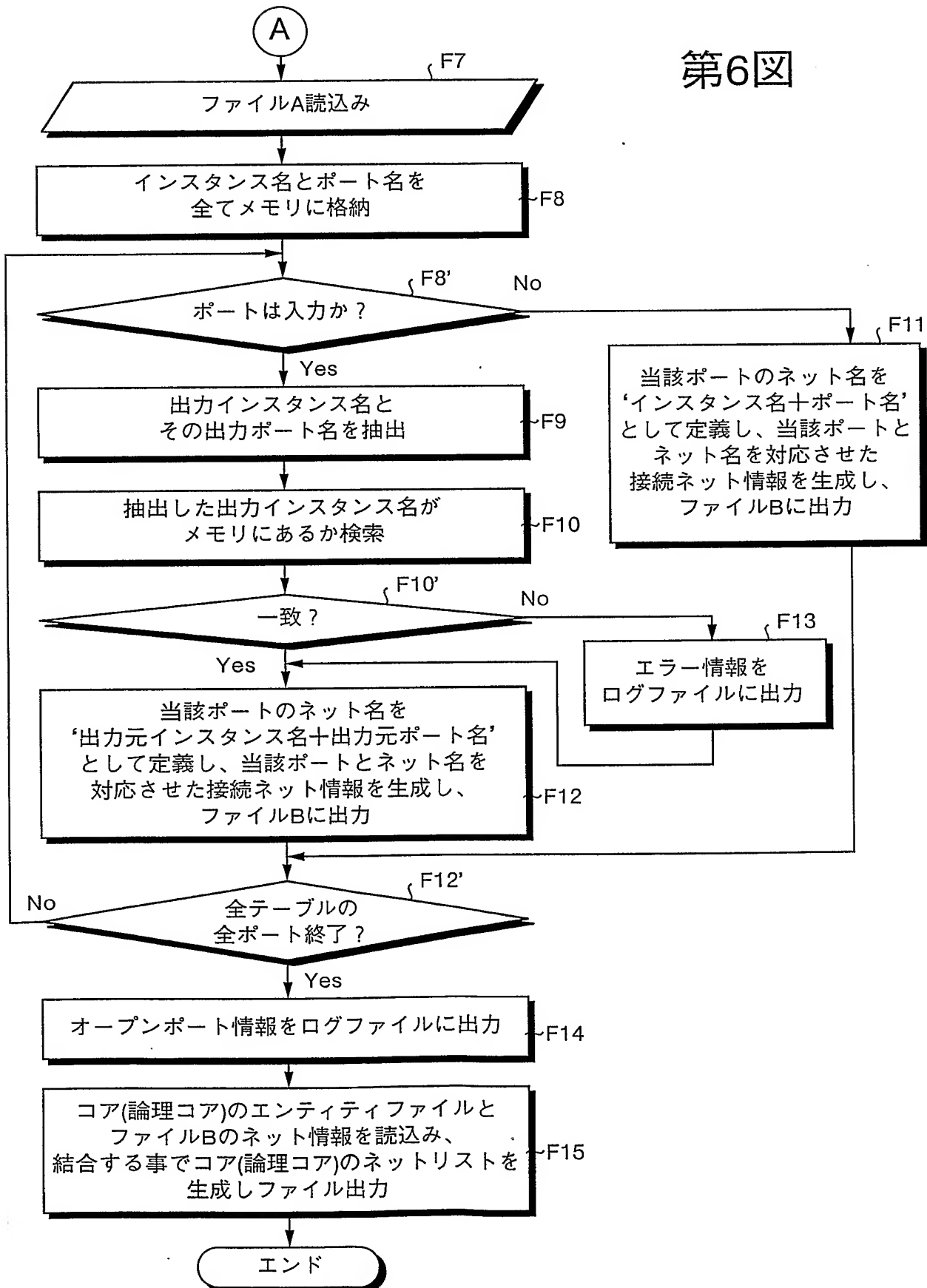
5/37

第5図

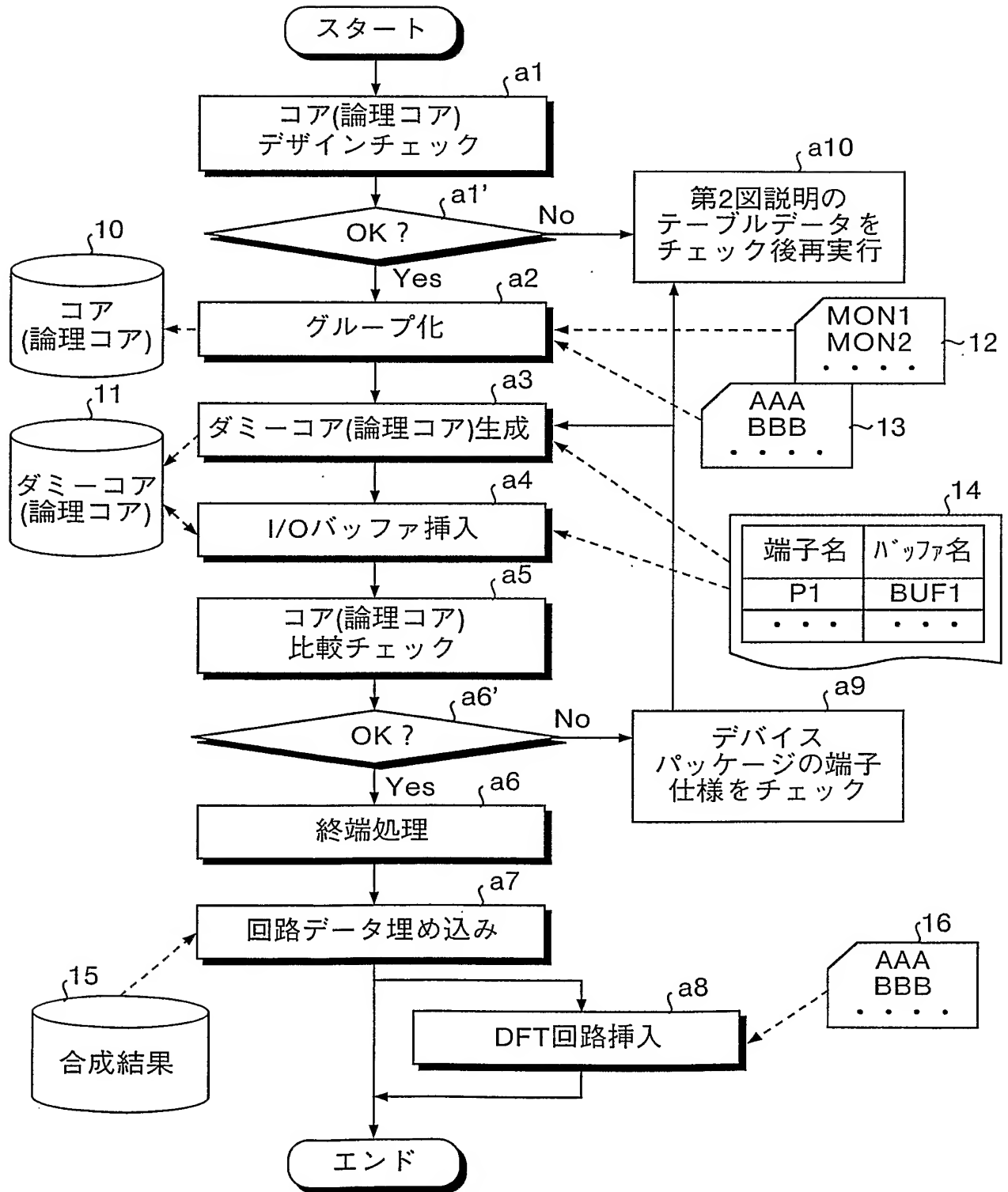


6/37

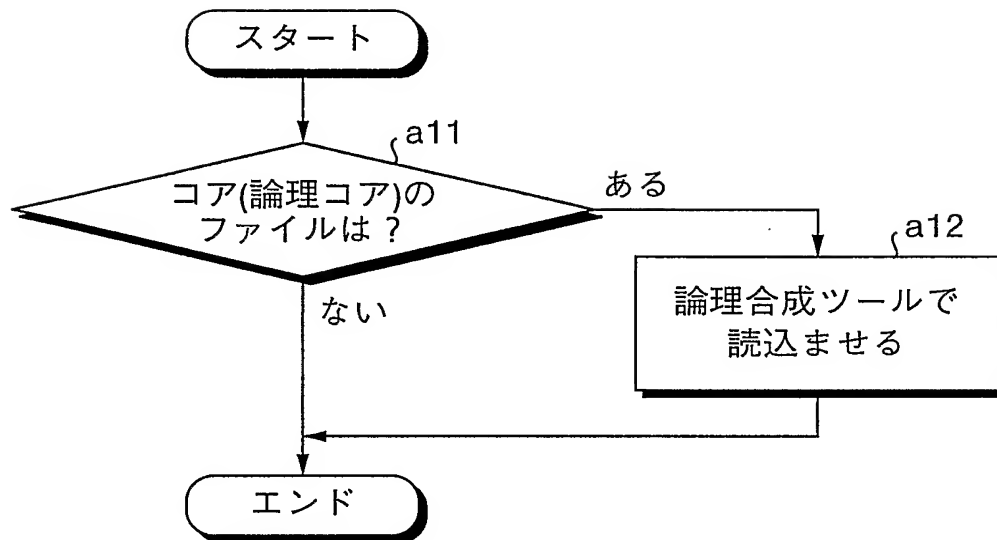
第6図



第7図

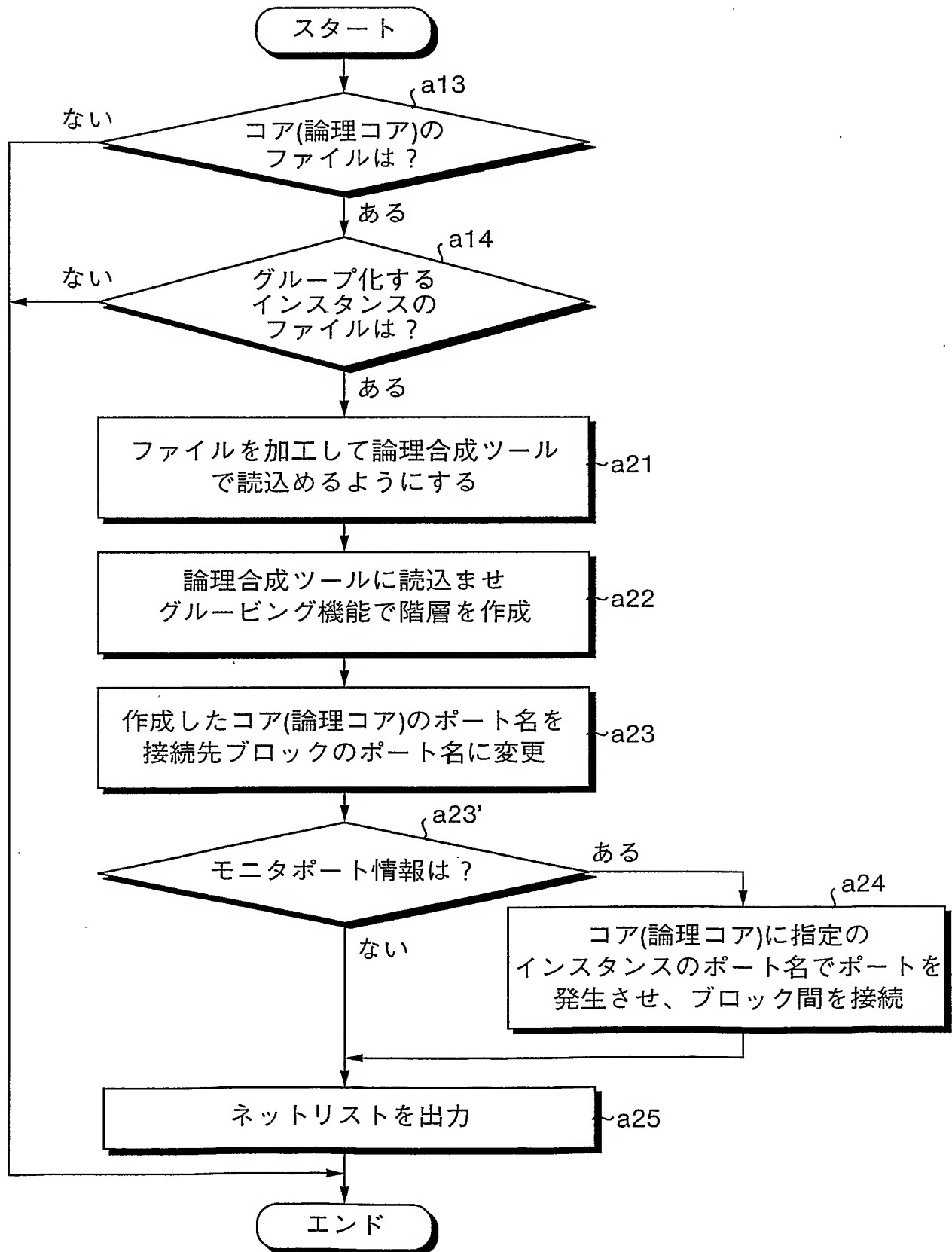


第8図



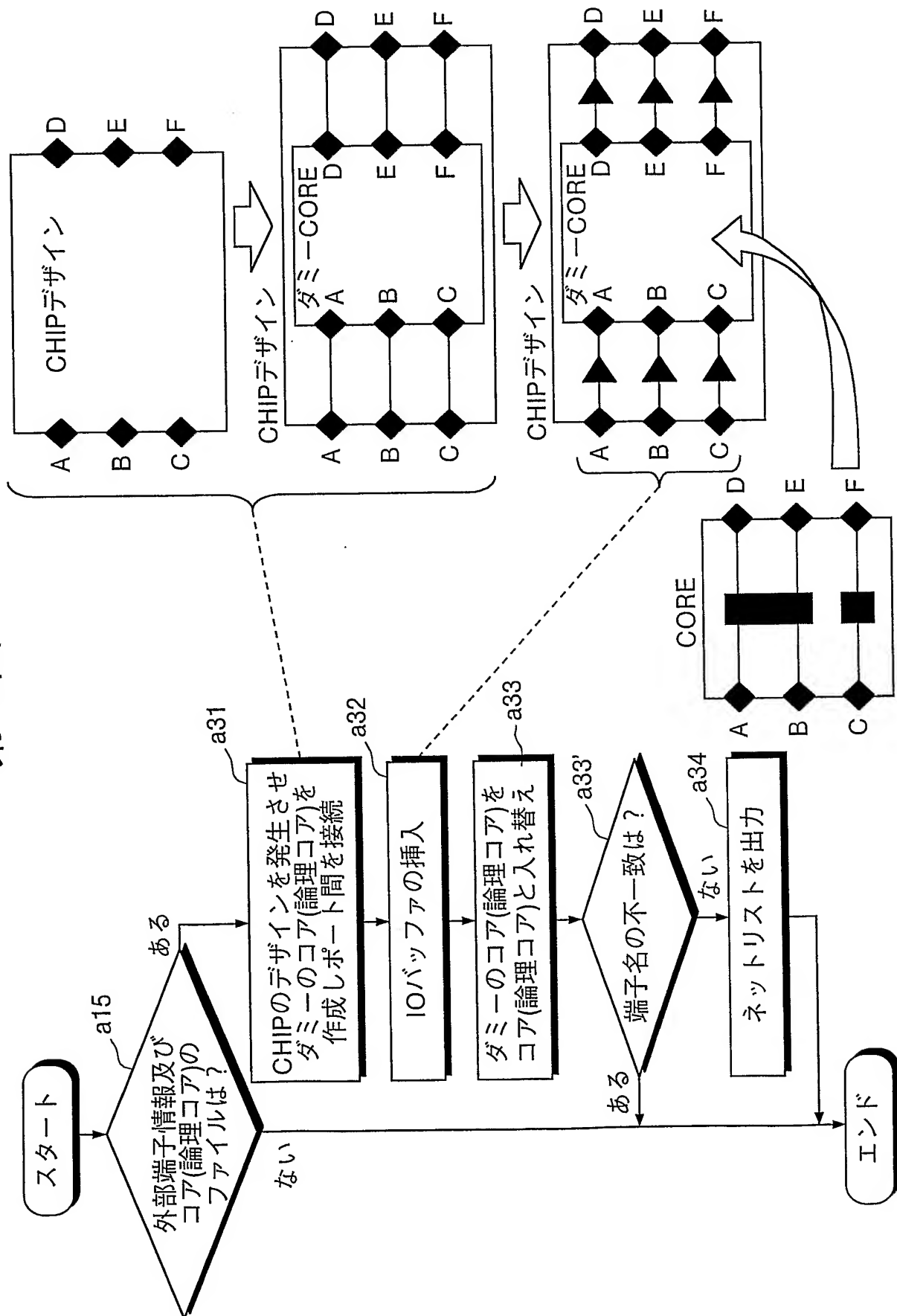
9/37

第9図

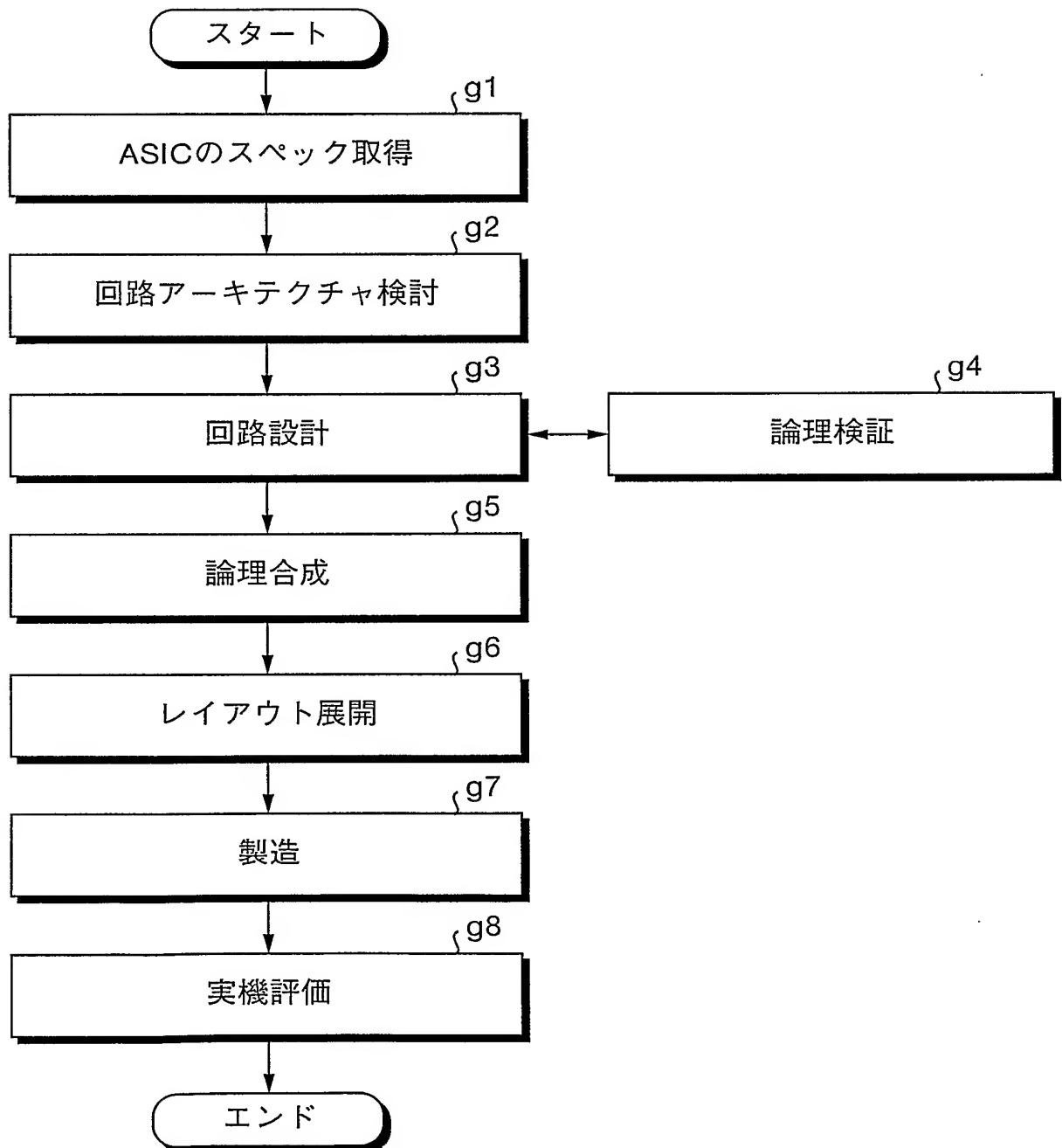


10/37

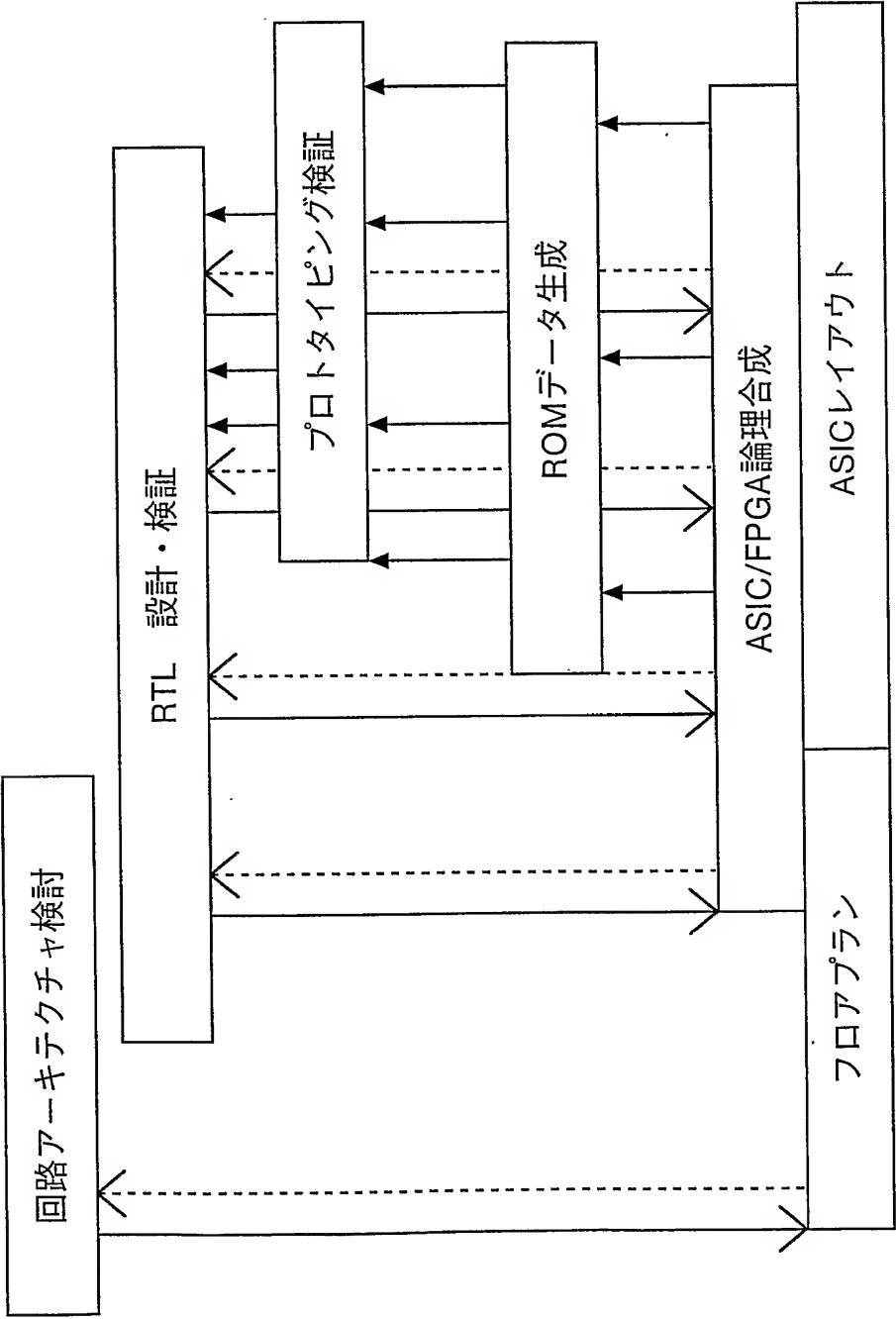
第10図



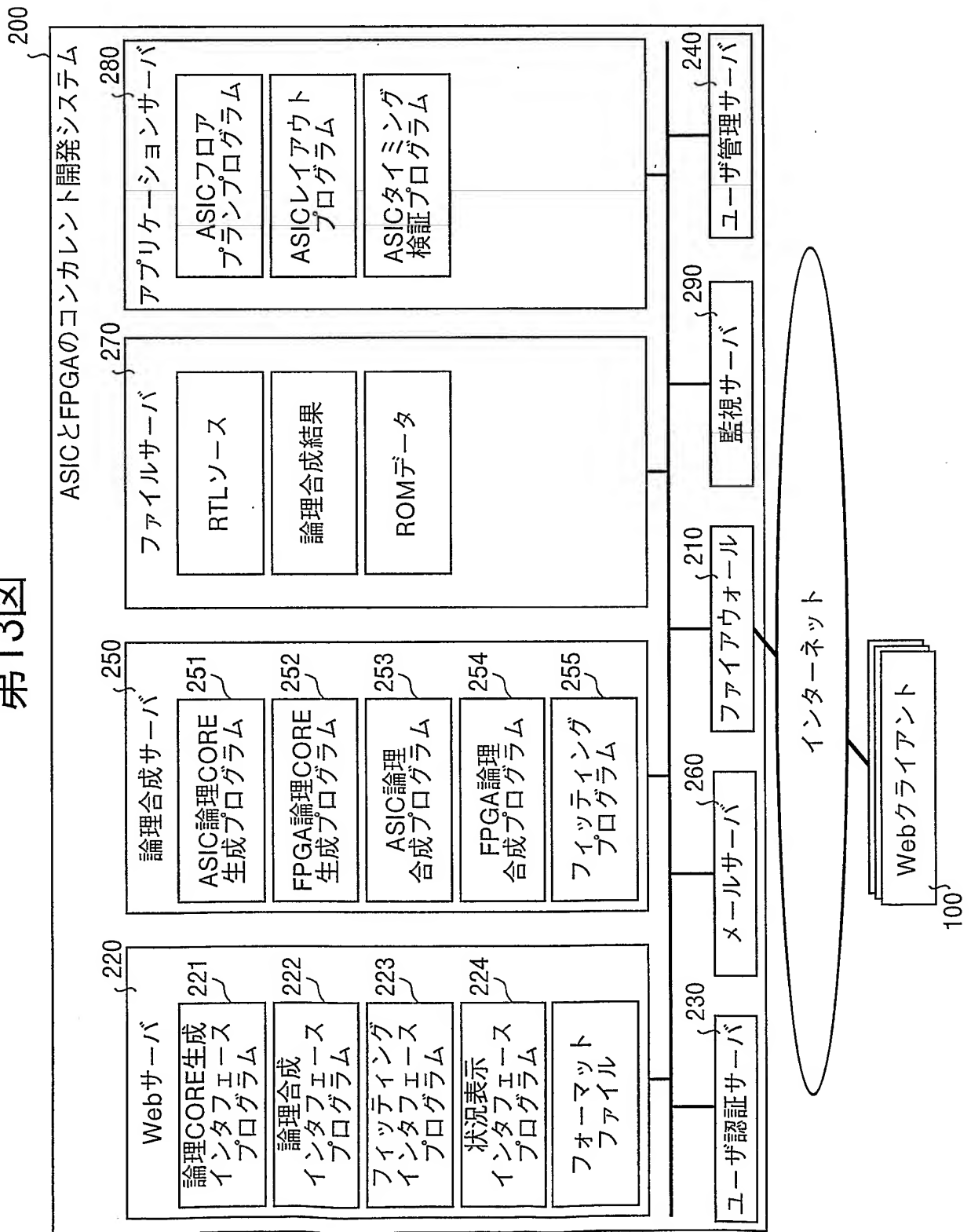
第11図



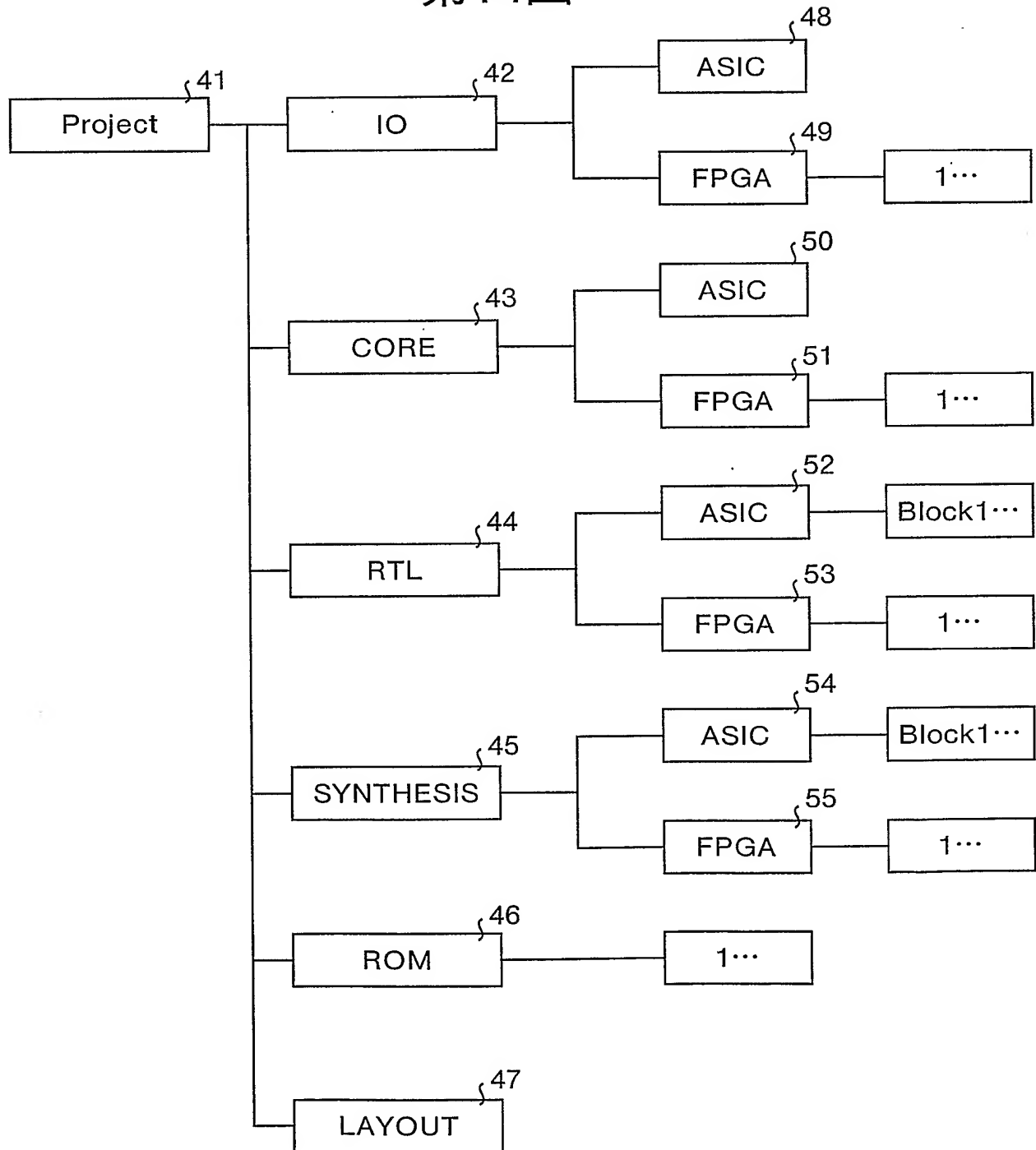
第12図



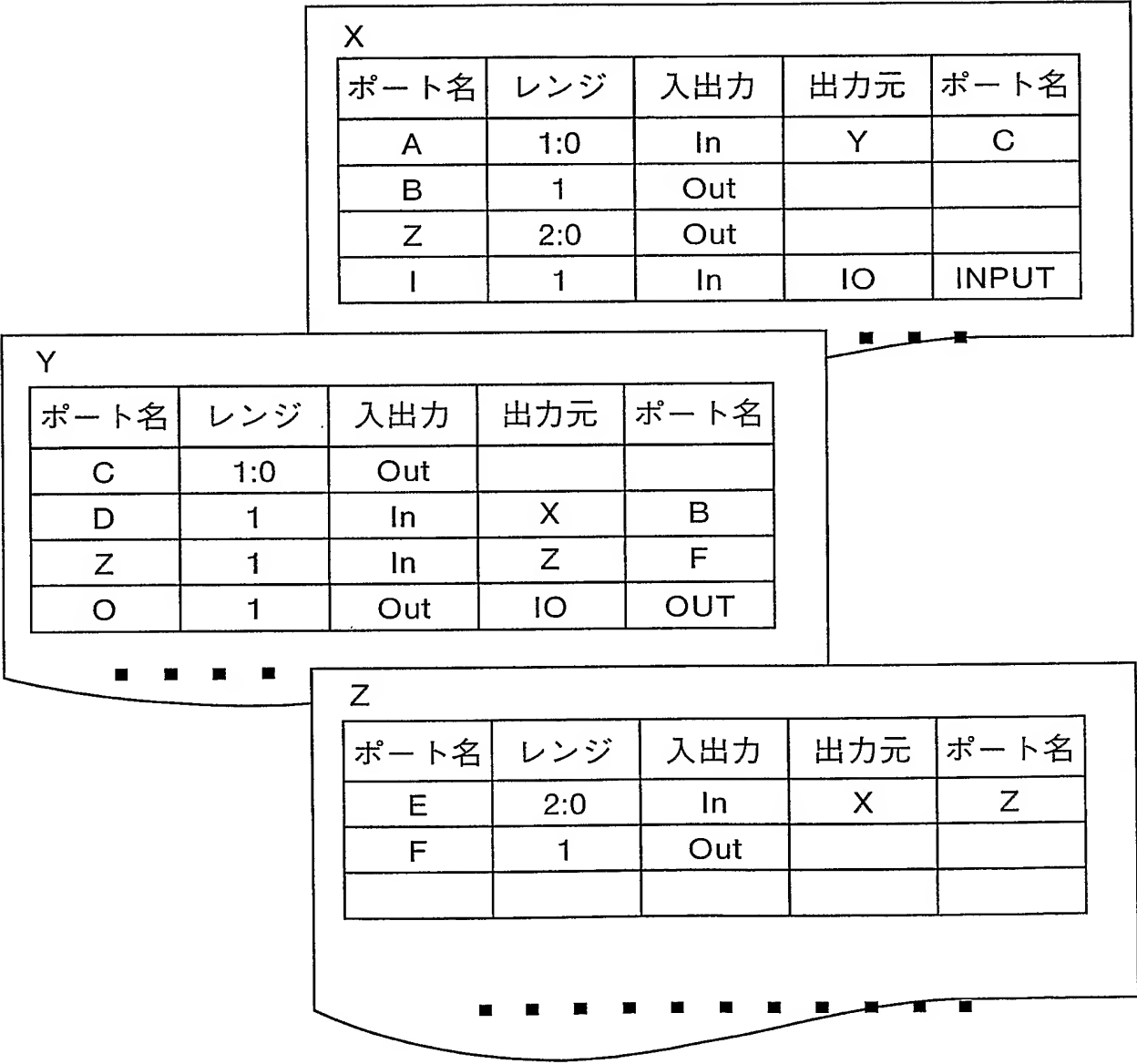
第13図

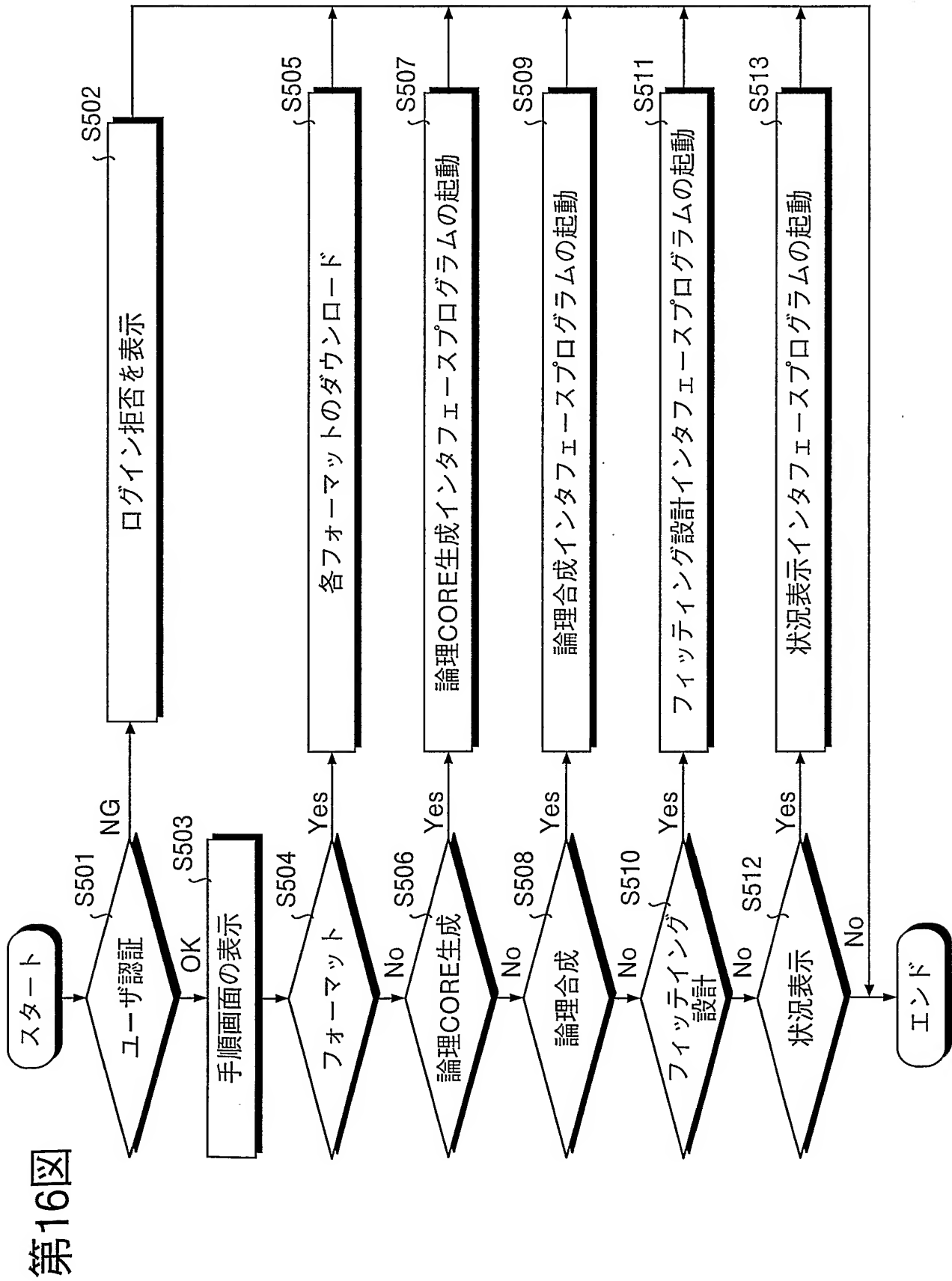


第14図



第15図





第17図

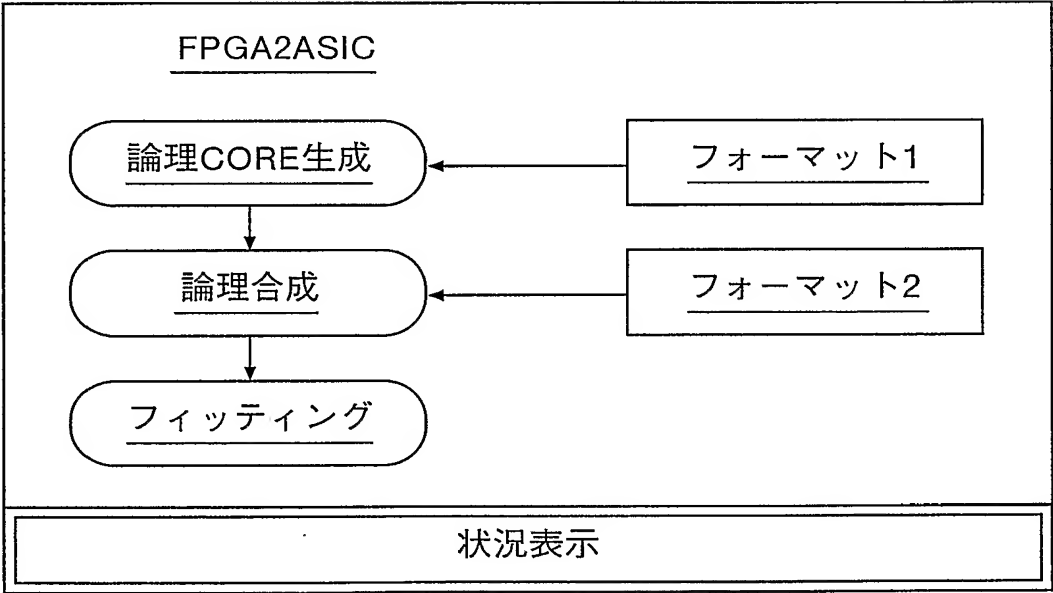
システムへのログイン

Name

Password

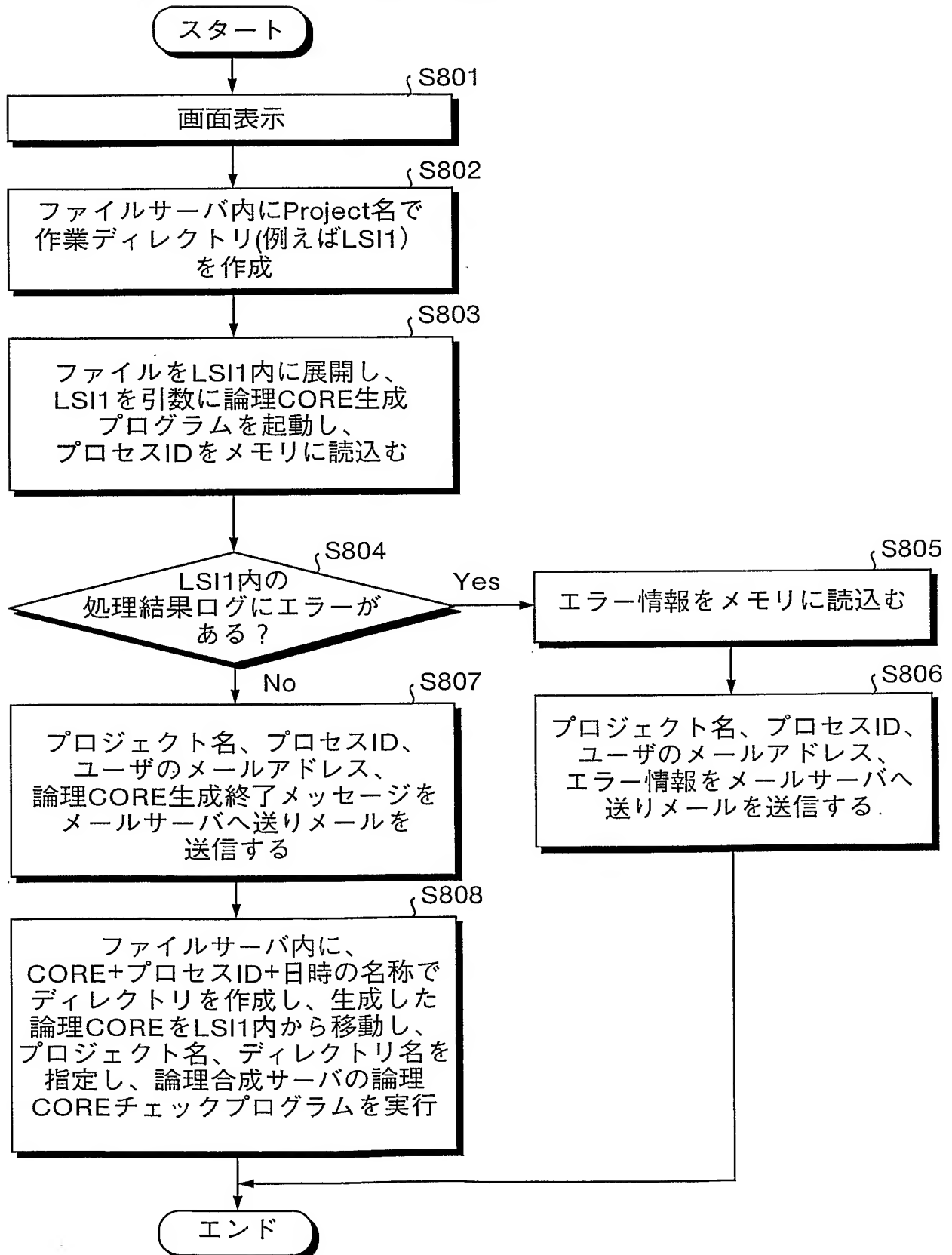
Login

第18図



19/37

第19図



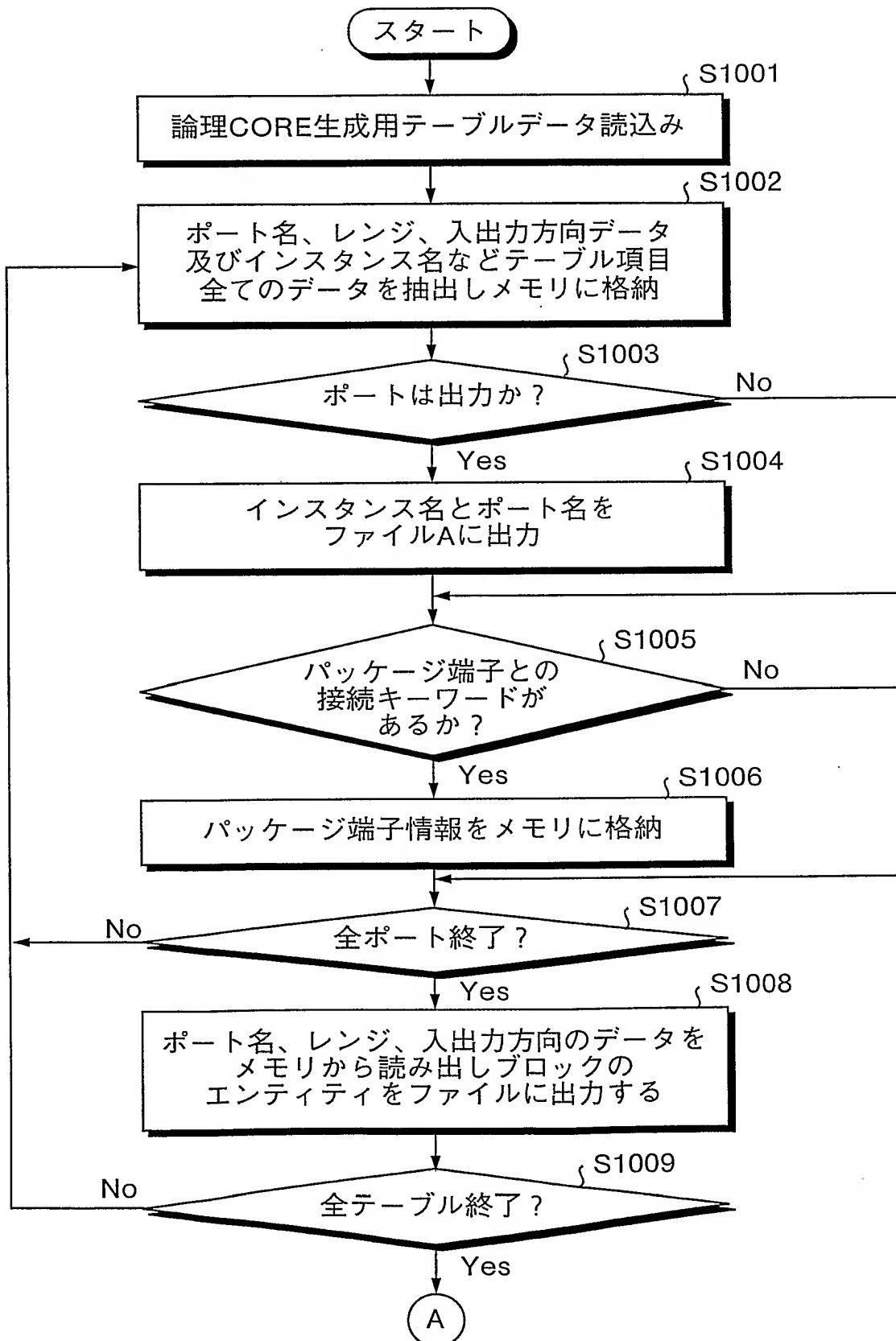
第20図

Project名

実行

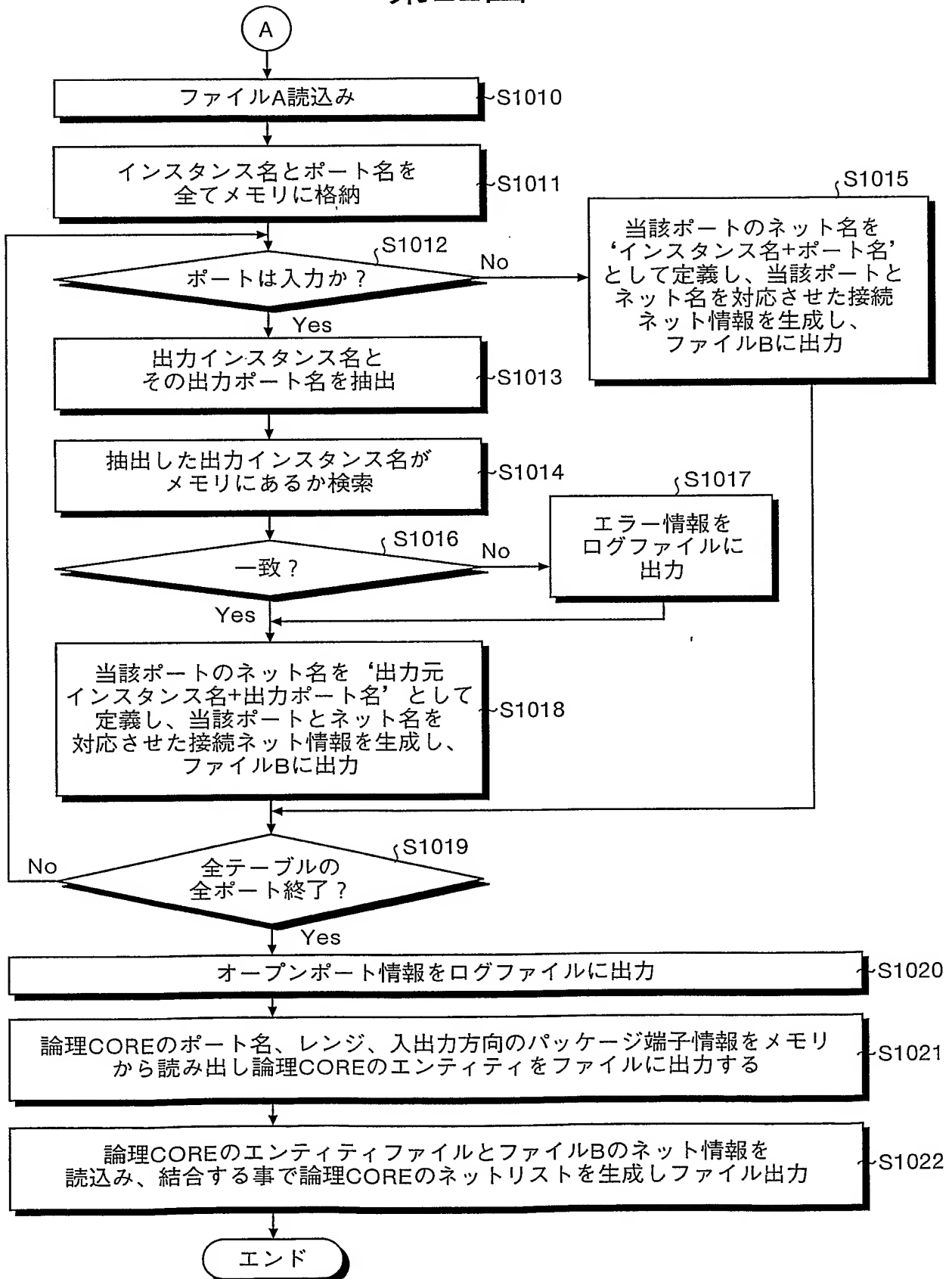
21/37

第21図

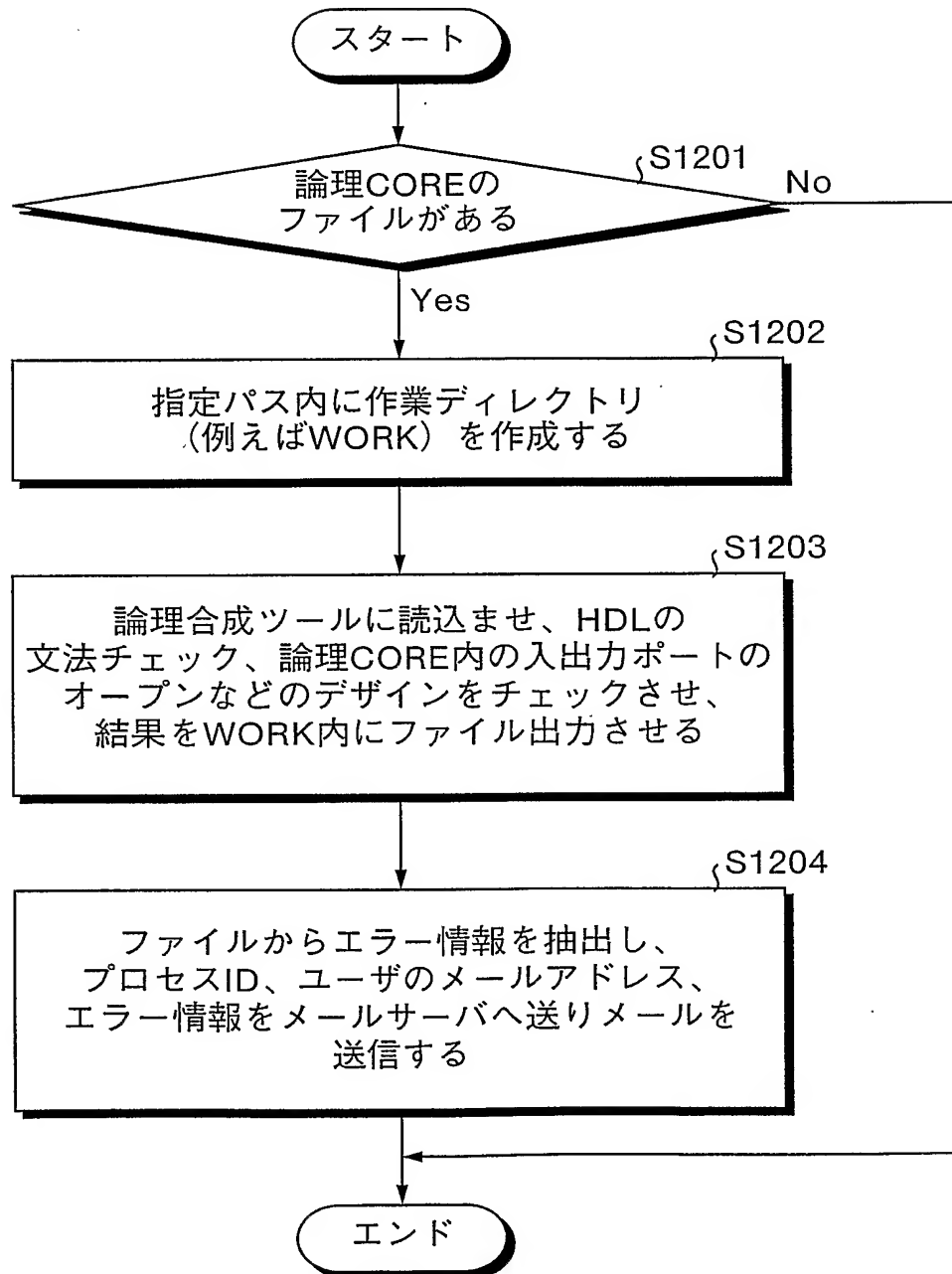


22/37

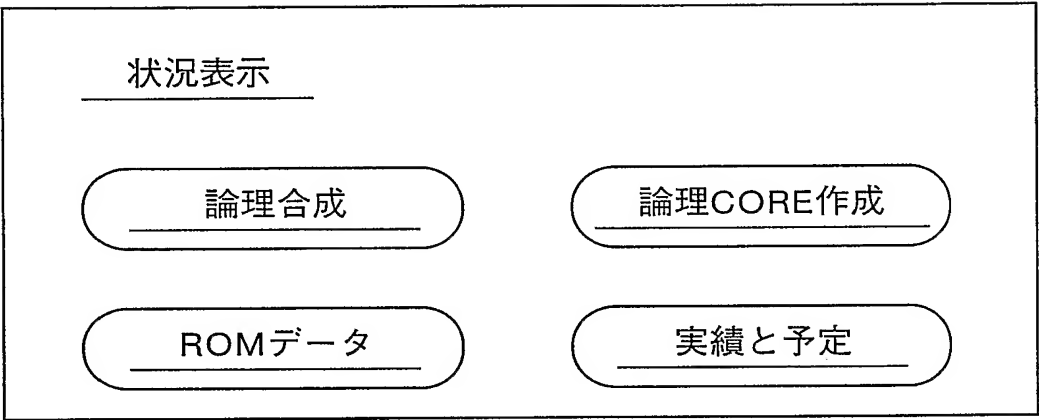
第22図



第23図



第24図



第25図

Project名

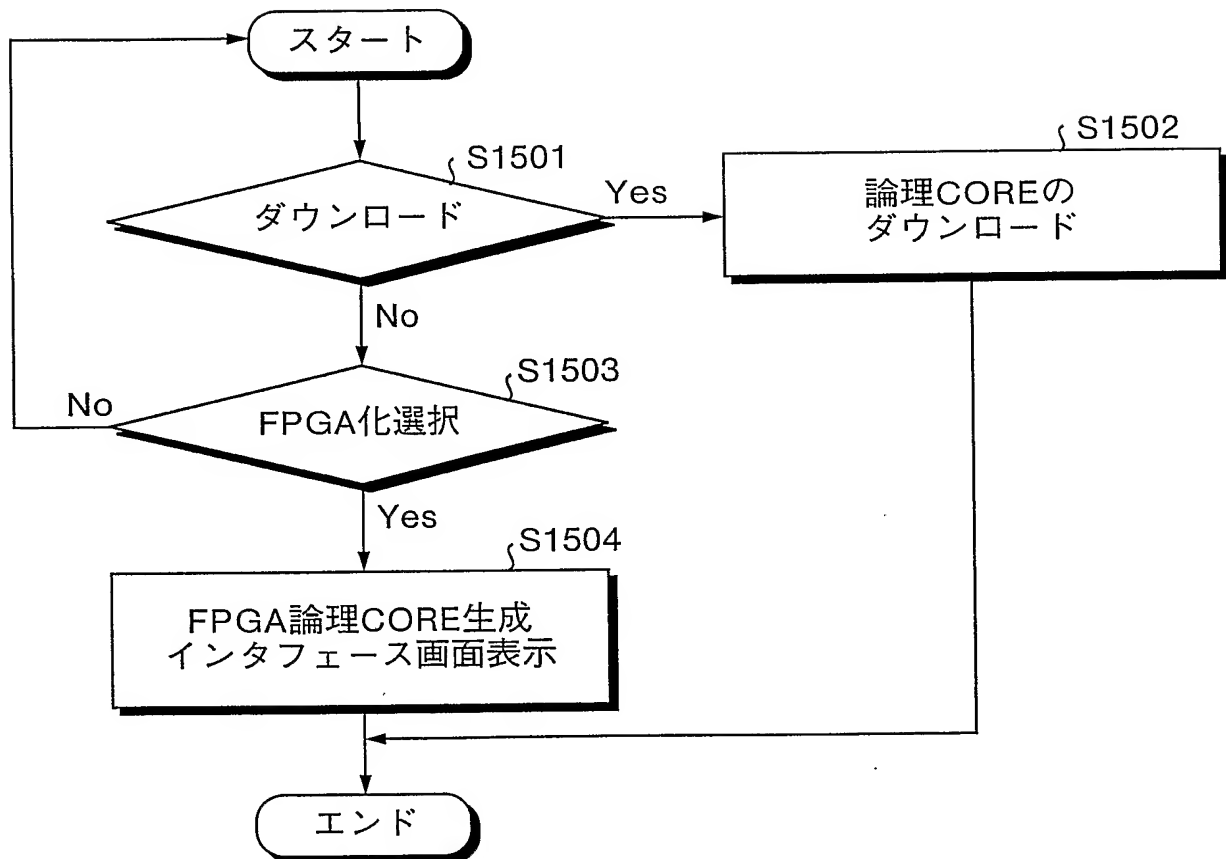
表示

CORE_0001yyyy.mm.dd
CORE_0002yyyy.mm.dd
■ ■ ■ ■ ■ ■ ■ ■ ■ ■
■ ■ ■ ■ ■ ■ ■ ■ ■ ■

ダウンロード

FPGA化

第26図

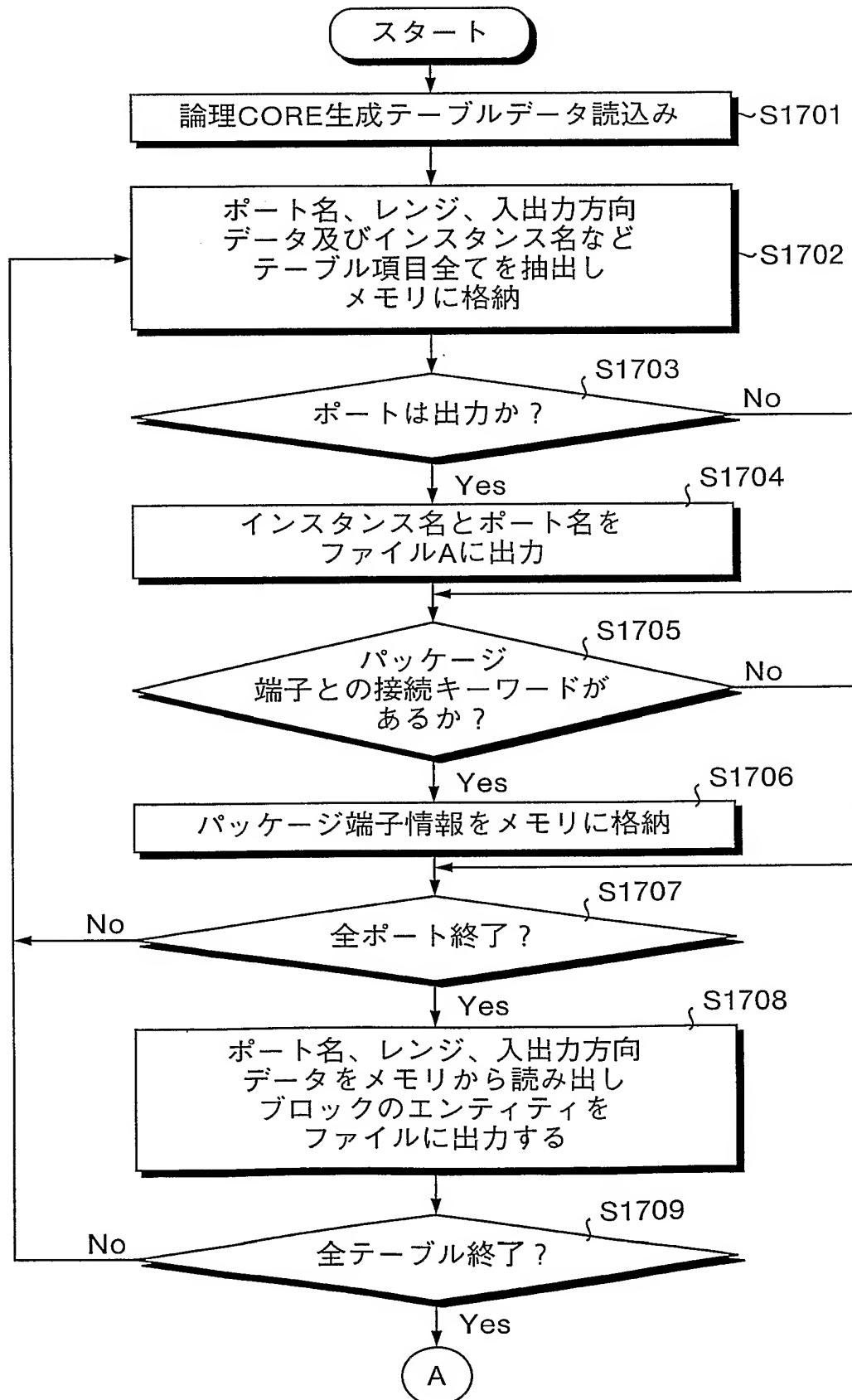


第27図

Project名	<input type="text"/>		
CORE_	<input type="text" value="0001yyyy.mm.dd"/>		
<div><div>BLOCK1 BLOCK2 BLOCK3</div><div>追加>></div><div>BLOCK2 BLOCK3</div></div>			
FPGA名:	<input type="text"/>	FPGA番号:	<input type="text"/>
<div>実行</div>			
<hr/>			
パッケージ名	:XXXXX,XXXXX,....		
IO使用率	:XX%		
<div>決定</div>			

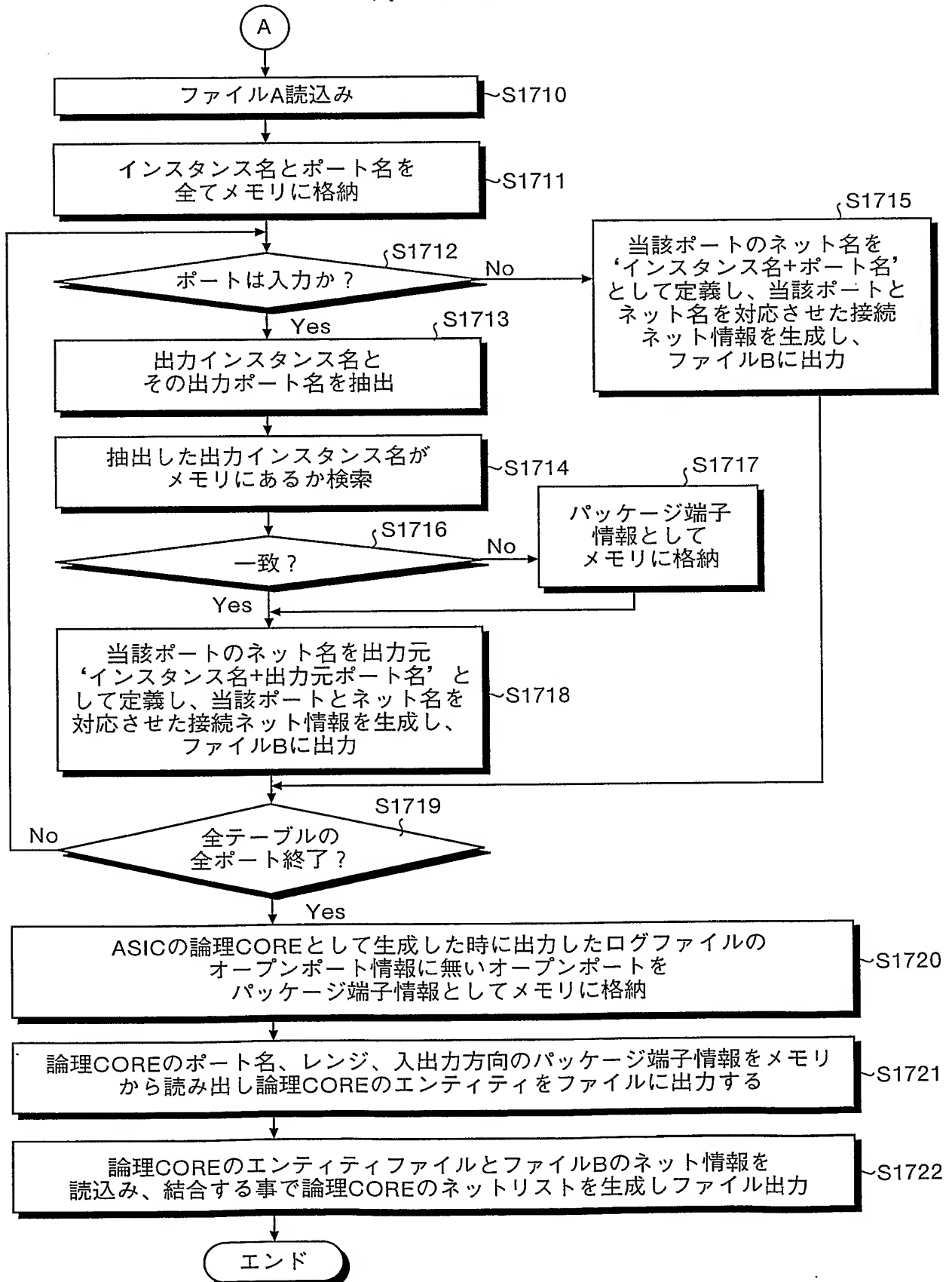
28/37

第28図



29/37

第29図

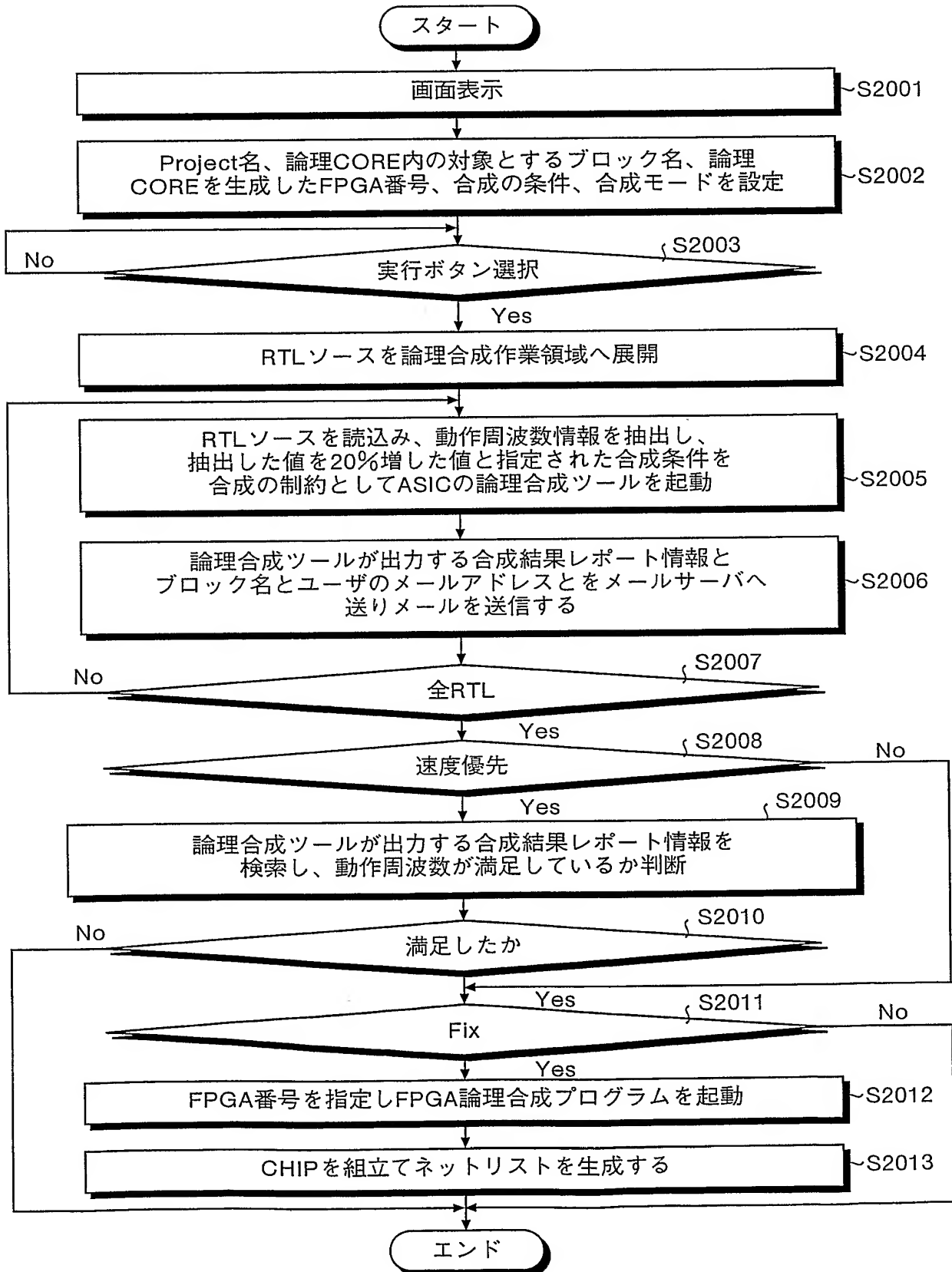


第30図

	IO数	ゲート規模	互換パッケージ
パッケージ名	XXX	GGGGG	ABCDEFGG,YYYYYYY,... ,...

31/37

第31図

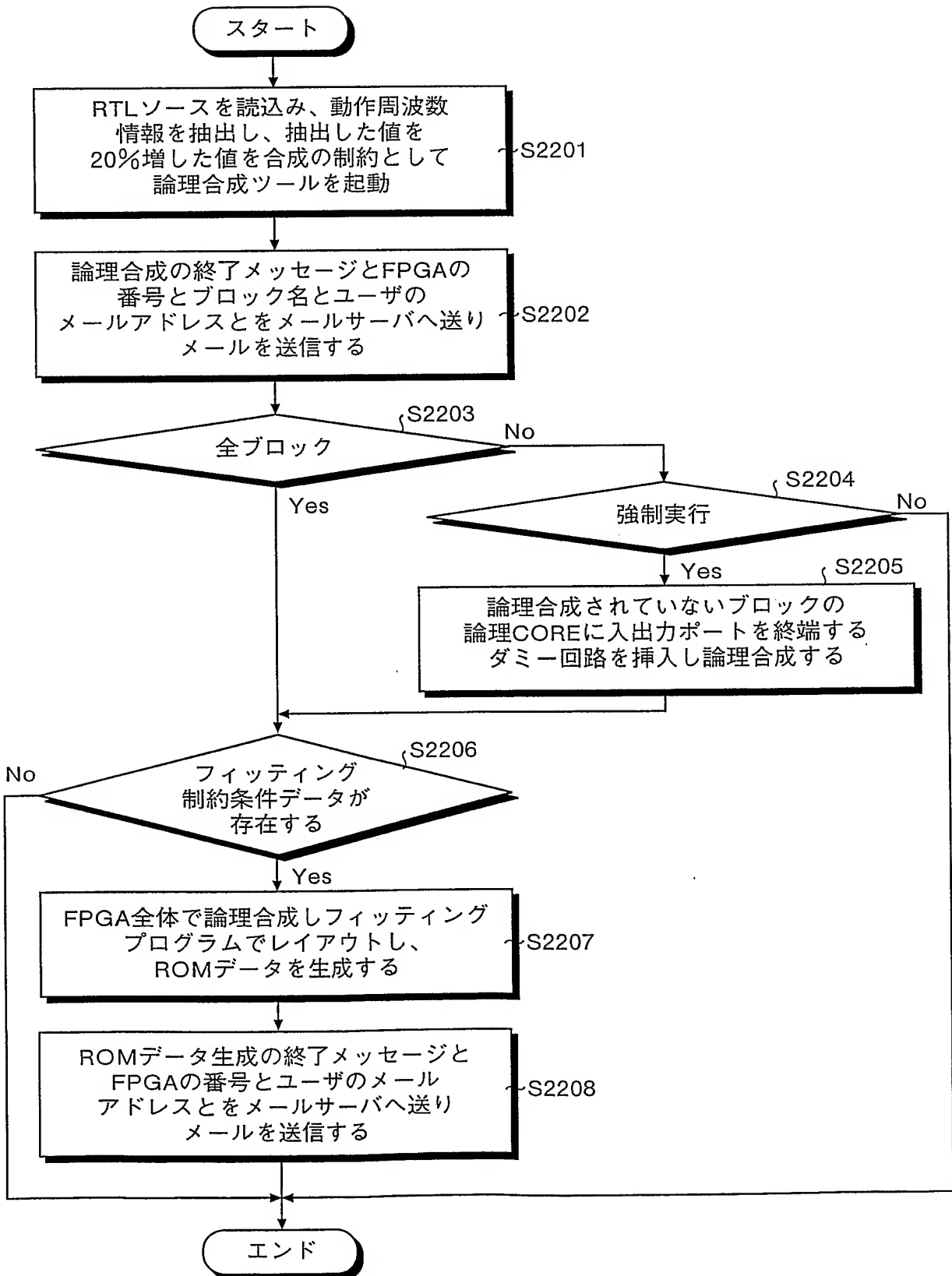


第32図

Project名	<input type="text"/>
対象論理CORE	<input type="text"/>
ブロック名	<input type="text"/>
FPGA番号	<input type="text"/>
<input type="checkbox"/> 面積重視	<input type="checkbox"/> 速度重視
<input type="checkbox"/> Debug	<input type="checkbox"/> Fix
<div>実行</div>	

33/37

第33図



第34図

Project名	:XXX		
FPGA番号	:1	2	3 ...
パッケージ名	:XXX	XXX	XXX
合成規模	:XXX	YYY	ZZZ
セル使用率	:50	70	75

第35図

FPGA No.	FPGA名		
1.	AAAAA	条件データ投入	実行
2.	BBBBB	条件データ投入	実行
3.	CCCCC	条件データ投入	実行
...
...

第36図

Project名：XXXX		
No.	FPGA名	日付
1	AAAA	yyyy/mm/dd
2	BBBB	■ ■ ■ ■ ■ ■ ■ ■
3	■ ■ ■ ■ ■ ■	■ ■ ■ ■ ■ ■ ■ ■
■ ■		

第37図

Project名:XXXXXX		
	予定日	実績日
1st RTL	yyyy/mm/dd1	yyyy/mm/dd2
.....		
.....		
反映開始日	yyyy/mm/dd3	yyyy/mm/dd3
反映開始日	yyyy/mm/dd4	
Sign Off	yyyy/mm1/dd	

論理Fix

変更

yyyy/mm2/dd

第38図

ブロック名	所要時間
Block1	1h
Block2	5h
.....	
.....	

INTERNATIONAL SEARCH REPORT

International application No.

PCT/JP03/04787

A. CLASSIFICATION OF SUBJECT MATTER

Int.Cl⁷ G06F17/50, H03K19/173, H01L21/82

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

Int.Cl⁷ G06F17/50, H03K19/173, H01L21/82

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

JSTplus FILE(JOIS)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	JP 2000-90142 A (Fujitsu Ltd.), 31 March, 2000 (31.03.00), Full text (Family: none)	1-16
Y	Tomoaki KOGA et al., "Mega Gate ASIC no Implement Sekkei", The Institute of Electronics, Information and Communication Engineers Gijutsu Kenkyu Hokoku, The Institute of Electronics, Information and Communication Engineers, 22 September, 1998 (22.09.98), Vol.98, No.287, pages 71 to 77 (VLD98-54)	1-16
Y	JP 8-194725 A (Fujitsu Ltd.), 30 July, 1996 (30.07.96), Full text (Family: none)	1-16

☒ Further documents are listed in the continuation of Box C.
 ☐ See patent family annex.

* Special categories of cited documents:	"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A" document defining the general state of the art which is not considered to be of particular relevance	"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E" earlier document but published on or after the international filing date	"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&" document member of the same patent family
"O" document referring to an oral disclosure, use, exhibition or other means	
"P" document published prior to the international filing date but later than the priority date claimed	

Date of the actual completion of the international search
01 July, 2003 (01.07.03)Date of mailing of the international search report
15 July, 2003 (15.07.03)Name and mailing address of the ISA/
Japanese Patent Office

Authorized officer

Facsimile No.

Telephone No.

INTERNATIONAL SEARCH REPORT

International application No.

PCT/JP03/04787

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
P, X	Koichi HANDA et al., "FPGA/PLD-kan Kessen Check Shuho", The Institute of Electronics, Information and Communication Engineers Gijutsu Kenkyu Hokoku, The Institute of Electronics, Information and Communication Engineers, 28 November, 2002 (28.11.02), Vol.102, No.479, pages 115 to 120 (VLD2002-101)	1-16

INTERNATIONAL SEARCH REPORT

International application No.

PCT/JP03/04787

Box I Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. ☐ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. ☐ Claims Nos.:
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box II Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

Performing logical synthesis in an integrated circuit design is a conventional technique.

Moreover, provision of an I/O buffer in an integrated circuit and using elements (including the I/O buffer) corresponding to the device technology belong to technical common sense.

Accordingly, the technical feature common to the inventions of claims 1-10, the inventions of claims 11-15, and the invention of claim 16 makes no contribution over the prior art and cannot be a special technical feature within the meaning of PCT Rule 13.2, second sentence.

(continued to extra sheet)

1. ☒ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.

2. ☐ As all searchable claims could be searched without effort justifying an additional fee, this Authority did not invite payment of any additional fee.

3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. ☐ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

Remark on Protest ☐ The additional search fees were accompanied by the applicant's protest.
☒ No protest accompanied the payment of additional search fees.

INTERNATIONAL SEARCH REPORT

International application No.

PCT/JP03/04787

Continuation of Box No.II of continuation of first sheet(1)

Consequently, the inventions of claims 1-10, the inventions of claims 11-15, and the invention of claim 16 do not satisfy the requirement of unity of invention.

A. 発明の属する分野の分類 (国際特許分類 (IPC))

Int. Cl. 7 G06F17/50, H03K19/173, H01L21/82

B. 調査を行った分野

調査を行った最小限資料 (国際特許分類 (IPC))

Int. Cl. 7 G06F17/50, H03K19/173, H01L21/82

最小限資料以外の資料で調査を行った分野に含まれるもの

国際調査で使用した電子データベース (データベースの名称、調査に使用した用語)

JSTplusファイル (JOIS)

C. 関連すると認められる文献

引用文献の カテゴリー*	引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	関連する 請求の範囲の番号
Y	JP 2000-90142 A (富士通株式会社) 2000. 03. 31, 全文, (ファミリーなし)	1-16
Y	古賀智昭, 外6名, 'メガゲートASICのインプリメント設計', 電子情報通信学会技術研究報告, 電子情報通信学会, 1998. 09. 22, Vol. 98, No. 287, p. 71-77 (VLD98-54)	1-16
Y	JP 8-194725 A (富士通株式会社) 1996. 07. 30, 全文, (ファミリーなし)	1-16

☒ C欄の続きにも文献が列挙されている。☐ パテントファミリーに関する別紙を参照。

* 引用文献のカテゴリー

「A」特に関連のある文献ではなく、一般的技術水準を示すもの

「E」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの

「L」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献 (理由を付す)

「O」口頭による開示、使用、展示等に言及する文献

「P」国際出願日前で、かつ優先権の主張の基礎となる出願

の日の後に公表された文献

「T」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの

「X」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの

「Y」特に関連のある文献であって、当該文献と他の1以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの

「&」同一パテントファミリー文献

国際調査を完了した日

01. 07. 03

国際調査報告の発送日 15.07.03

国際調査機関の名称及びあて先

日本国特許庁 (ISA/JP)

郵便番号 100-8915

東京都千代田区霞が関三丁目4番3号

特許庁審査官 (権限のある職員)

早川 学



5H

9652

電話番号 03-3581-1101 内線 3531

C (続き) . 関連すると認められる文献		関連する 請求の範囲の番号
引用文献の カテゴリー*	引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	
PX	反田浩一，外 1 名，‘FPGA／PLD 間結線チェック手法’， 電子情報通信学会技術研究報告，電子情報通信学会，2002. 11. 28， Vol. 102，No. 479，p. 115-120 (VLD2002-101)	1-16

第Ⅰ欄 請求の範囲の一部の調査ができないときの意見（第1ページの2の続き）

法第8条第3項（PCT17条(2)(a)）の規定により、この国際調査報告は次の理由により請求の範囲の一部について作成しなかった。

1. ☐ 請求の範囲 _____ は、この国際調査機関が調査をすることを要しない対象に係るものである。つまり、
2. ☐ 請求の範囲 _____ は、有意義な国際調査をすることができる程度まで所定の要件を満たしていない国際出願の部分に係るものである。つまり、
3. ☐ 請求の範囲 _____ は、従属請求の範囲であってPCT規則6.4(a)の第2文及び第3文の規定に従って記載されていない。

第Ⅱ欄 発明の単一性が欠如しているときの意見（第1ページの3の続き）

次に述べるようにこの国際出願に二以上の発明があるとこの国際調査機関は認めた。

集積回路設計において、論理合成を行うことは慣用技術である。

また、集積回路において、I/Oバッファを設けること、及び、素子（I/Oバッファを含む。）をデバイス・テクノロジーに対応したものとすることは技術常識である。

してみれば、請求の範囲1-10に係る発明と、請求の範囲11-15に係る発明と、請求の範囲16に係る発明との間に共通する事項は、先行技術の域を出ないから、PCT規則13.2の第2文の意味において、特別な技術的特徴ではない。

したがって、請求の範囲1-10に係る発明と、請求の範囲11-15に係る発明と、請求の範囲16に係る発明とは、単一性の要件を満たしていない。

1. ☒ 出願人が必要な追加調査手数料をすべて期間内に納付したので、この国際調査報告は、すべての調査可能な請求の範囲について作成した。
2. ☐ 追加調査手数料を要求するまでもなく、すべての調査可能な請求の範囲について調査することができたので、追加調査手数料の納付を求めなかった。
3. ☐ 出願人が必要な追加調査手数料を一部のみしか期間内に納付しなかったため、この国際調査報告は、手数料の納付のあった次の請求の範囲のみについて作成した。
4. ☐ 出願人が必要な追加調査手数料を期間内に納付しなかったため、この国際調査報告は、請求の範囲の最初に記載されている発明に係る次の請求の範囲について作成した。

追加調査手数料の異議の申立てに関する注意

- ☐ 追加調査手数料の納付と共に出願人から異議申立てがあった。
- ☒ 追加調査手数料の納付と共に出願人から異議申立てがなかった。